Guilherme Brunel Zaffari

Um Modelo Bioinspirado para Localização e Mapeamento Simultâneos de Robôs Subaquáticos em Ambientes com a Influência da Correnteza

Brasil

2017

Guilherme Brunel Zaffari

Um Modelo Bioinspirado para Localização e Mapeamento Simultâneos de Robôs Subaquáticos em Ambientes com a Influência da Correnteza

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal do Rio Grande, como requisito parcial para a obtenção do grau de Mestre em Engenharia de Computação

Universidade Federal do Rio Grande

Centro de Ciências Computacionais

Programa de Pós-Graduação em Engenharia de Computação

Orientador: Prof. Dra. Silvia Silva da Costa Botelho

Coorientador: Prof. Dr. Paulo Lilles Jorge Drews Junior

Dr. Hendry Ferreira Chame

Brasil

2017

Ficha catalográfica

Z17m Zaffari, Guilherme Brunel.
Um modelo de bioinspirado para localização e mapeamento simultâneos de robôs subaquáticos em ambientes com a influência da correnteza / Guilherme Brunel Zaffari. – 2017.
104 p.

Dissertação (mestrado) – Universidade Federal do Rio Grande – FURG, Programa de Pós-graduação em Engenharia da Computação, Rio Grande/RS, 2017.
Orientadora: Drª. Silvia Silva da Costa Botelho.
Coorientador: Dr. Paulo Lilles Jorge Drews Junior.
Coorientador: Dr. Hendry Ferreira Chame.

1. Localização e mapeamento simultâneos 2. Algoritmo Bioinspirado 3. Ambiente Subaquático I. Botelho, Silvia Silva da Costa II. Drews Junior, Paulo Lilles Jorge III. Chame, Hendry Ferreira IV. Título.

CDU 004.896

Catalogação na Fonte: Bibliotecário Me. João Paulo Borges da Silveira CRB 10/2130

UNIVERSIDADE FEDERAL DO RIO GRANDE

Centro de Ciências Computacionais

Programa da Pós-Graduação em Computação

Curso de Mestrado em Engenharia de Computação

DISSERTAÇÃO DE MESTRADO

Um Modelo Bioinspirado para Localização e Mapeamento Simultâneos de Robôs Subaquáticos em Ambientes com a Influência de Correnteza

Guilherme Brunel Zaffari

Banca examinadora

Prof. Dr. Armando Alves Neto

Prof. Dr. Héctor Cancela

Prof. Dr. Rodrigo Zelir Azzolin

Profa. Dra. Silvia Silva da Costa Botelho
Orientadora

Prof. Dr. Hendry Ferreira Chame
Coorientador

Prof. Dr. Paulo Lilles Jorge Drews Jr.
Coorientador

Este trabalho é dedicado a todas as pessoas
que de alguma forma contribuíram para a realização da dissertação.

Agradecimentos

Primeiramente gostaria de agradecer a Prof. Dr^a Silvia Botelho por ter me orientado durante toda a realização deste trabalho. Também ao Prof. Dr. Paulo Drews e ao Dr. Hendry Chame que no papel de co-orientadores contribuíram na orientação com ideias, discussões nas reuniões relativas ao projeto.

Agradeço a minha família, em especial meus pais e meu irmão por compreenderem a ausência nos feriados, aniversários e outras ocasiões. Aos meus pais agradeço o incentivo aos meus estudos desde à graduação até a escolha de realizar o mestrado. Ao meu irmão pelas conversas descontraídas.

Também agradeço as amizades que criei e as que eu fortaleci durante esta etapa da minha vida. Aos amigos da sala B-06 ou antigo "aquário"as quais contribuíram de forma significativa para que este trabalho fosse realizado: Amanda , Jusoan, Everson, Cristiano, Joel, Felipe Codevilla, Rômulo. Em especial gostaria de agradecer Matheus Machado e Lucas Longaray pela ajuda no desenvolvimento dos datasets utilizados neste trabalho.

Aos amigos do Cassino: Bianca, Mauricio, Daiane, Jonas, Thaisa, Douglas, Guilherme, Átila e Luciane. Gostaria de agradecer pelas conversas, conselhos e pelas saídas mesmo que para tomar um chimarrão.

Gostaria de agradecer ao IBP - Instituto Brasileiro de Petróleo, Gás e Biocombustíveis pelo apoio financeiro a pesquisa por meio do Programa de Recursos Humanos da ANP para o Setor Petróleo e Gás - PRH-ANP/MCT. Também gostaria de agradecer ao Programa de Recursos Humanos da FURG – PRH-27 e a Prof. Dr^a Maria Isabel pelo empenho e dedicação pelo programa. A todos até aqui mencionados, Muito Obrigado.

"Carry on my wayward son,
There'll be peace when you are done
Lay your weary head to rest
Don't you cry no more.
(Kansas - Carry On My Wayward Son)

Resumo

Neste trabalho é apresentado um modelo bioinspirado para o problema de localização e mapeamento simultâneos para robôs utilizados em ambientes subaquáticos com a influência da correnteza. Devido as suas características específicas, os desafios encontrados e os sensores utilizados nas aplicações em ambientes subaquáticos são apresentados. A abordagem bioinspirada é apresentada, onde são utilizadas para o seu desenvolvimento o comportamento das células *place cells*, *head direction cells* e *grid cells* as quais são encontradas nos cérebros dos mamíferos. Para avaliar o desempenho do modelo utilizam-se simulações do ambiente subaquático com a influência da correnteza desenvolvidas através do simulador Gazebo em conjunto com plugins do *UUV Simulator*. Os mapas da trajetória realizado pelo robô são obtidos utilizando o modelo proposto e comparados com o *ground thruth* (valor de referência) e *dead reckoning* (navegação estimada) obtidos durante a simulação.

Palavras-chaves: Localização e mapeamento simultâneos, Algoritmo Bioinspirado, Ambiente Subaquático.

Abstract

This work presents a bio-inspired model to solve the problem of simultaneous localization and mapping for robots in underwater environments, under the influence of water currents. Due to specific characteristics of sensory equipments and robot configurations used in underwater robot tasks, the challenges encountered are presented through applications. Thus, a bio-inspired approach is presented where behavior is modeled based on the concept of celular organizations found in the mammalian brain such as place cells, head direction cell cells, and grid cells. In order to evaluate the performance of the model under the influence of water currents. the navigation task is simulated in the Gazebo simulator and the UUV Simulator plug-ins. Trajectory maps generated by the proposed method are compared with the *ground truth* and *dead reckoning*, obtained during the simulation.

Key-words: Simultaneous localization and mapping. bio-inspired algorithm. Underwater environment.

Lista de ilustrações

Lista de tabelas

Lista de Abreviaturas e Siglas

AHRS *Attitude and Heading Reference System*

AUV *Autonomous Underwater Vehicle*

ASKF *Augmented State Kalman filter*

ADCP *Acoustic Doppler Current Profiler*

BOW *Bag Of Words*

CANN *Continuous Attractor Neural Network*

DGPS *Differential Global Positioning System*

DVL *Doppler Velocity Log*

EKF *Extended Kalman Filter*

ESEIF *Exactly Sparse Extended Information Filter*

FLS *Forward-Looking Sonar*

GPS *Global Positioning System*

IMU *Inertial Measurement Unit*

LASER *Light Amplification by Stimulated Emission of Radiation*

LBL *Long Base Line System*

MSIS *Mechanical Scanning Imaging Sona*r

RANSAC *Random Sample Consensus*

ROS *Robot Operating System*

ROV *Remotely Operated Underwater Vehicle*

SBL *Short Baseline Positioning System*

SIFT *Scale Invariant Feature Transform*

SLAM Localização e Mapeamento Simultâneos

SOM *Self-Organizing Map*

SONAR *Sound Navigation and Ranging*

SURF *Speeded Up Robust Features*

USBL *Ultra Short Baseline Positioning System*

UWSIM *Underwater Simulator*

Lista de símbolos

ε_r	Matriz de pesos excitatória
$rr_{x',y',\theta'}$	Cálculo da Excitação de cada Neurônio
$\beta r_{ix'y'\theta'}$	Matriz Armazenada de conexões
$e_r i$	Experiência criadas no mapa
S_r	Taxa de similaridade das experiências
tr_{ij}	Transição entre duas experiências
er_j	Nova experiência do mapa
Δpr^i	Erro entre duas experiências.
ϕ	Rotação em torno do eixo x.
θ	Rotação em torno do eixo y.
ψ	Rotação em torno do eixo z.
$R_{similar}$	Taxa de comparação dos histogramas
l	*Local View*.
L	Conjunto de todas as *Local Views* criadas.
$r_{x',y',z'}$	Taxa de excitação da rede.
ϵ	Valor do peso sináptico entre dois neurônios.
$B_{ix',y',z'}$	Peso sináptico relativo a excitação externa do sistema.
e_i	Nodo de experiência.
p_i	Posição espacial no mapa de experiência.
R^i	Ativação da Place Cell.
S^i	Taxa de ativação de uma experiência.
α_r	Constante de ponderação da taxa proveniente da Place Cell.
α_L	Constante de ponderação da taxa proveniente da Local View.

Δp^{ij}	Deslocamento entre a experiência i e a experiência j.
N_f	Número de conexões da experiência atual para as outras.
N_t	Número de conexões de outras experiências para a atual.
α	Constante de correção do mapa
h_i^p	Ativação de cada *Place Cell*
C^P	Número das ligações entre todas as *Place Cell*
w_{ij}^{RC}	Peso sináptico recorrente
r_i^p	Taxa de ativação da *Place Cell i*
I_i^V	Entrada visual da Rede
w_{ijklf}^{WC}	Peso das conexões idiotéticas
r_j^{-p}	*Trace Rule*
η	Contribuição da taxa de disparo da *Trace Rule*
r_k^{WCh}	Taxa de ativação da orientação da correnteza
r_l^{wCv}	Taxa de velocidade da correnteza
r_f	Taxa de ativação da orientação do robô
$C^{P\times WCh\times WCv}$	Número de ligações entre todos os neurônios da rede.

Sumário

1 Introdução

Atualmente os ambientes subaquáticos ainda são considerados praticamente inexplorados (NOAA, 2014). Novos recursos biológicos e minerais são encontrados no fundo do oceano, fazendo com que sua exploração seja considerada uma etapa crucial para novas descobertas. A indústria petrolífera pode ser considerada uma das principais investidoras em tecnologias para exploração, inspeção e manipulação em meios subaquáticos. Estes investimentos ocorrem devido ao aumento da demanda dos derivados do petróleo no mundo moderno, e dos recursos tornarem-se cada vez mais difíceis de serem encontrados em ambientes terrestres e águas rasas, ocasionando a exploração e extração para águas profundas.

As últimas décadas no Brasil foram marcadas pela busca da autossuficiência na produção de petróleo, que obteve sucesso graças aos avanços tecnológicos alcançados na exploração e operação em águas profundas (RIBEIRO, 2012). No Brasil, a descoberta das reservas do pré-sal na bacia de Santos tornaram-se um marco para a indústria. Através destas descobertas, os desafios aliados a extração do petróleo cresceram, principalmente pela dificuldade de acesso aos locais de perfuração, onde o controle e a supervisão da extração se dará por meio de computadores, sensores e robôs (BRASILIS, 2012).

Considerando as limitações humanas no ambiente subaquático e o avanço nos componentes eletrônicos, estão sendo desenvolvidos robôs não tripulados para realizar operações nos ambientes subaquáticos, os quais são divididos em *Remotely Operated Underwater Vehicle* (ROV) (veículo submarino operado remotamente) e *Autonomous Underwater Vehicle* (AUV) (veículos autônomos subaquáticos).

Os ROVs recebem energia e trocam informações com o sistema de controle na superfície através de um cabo umbilical. A partir do sistema de controle, o operador pode planejar tarefas ou utilizar uma interface de controle remoto (*joystick*) para manobrar diretamente o veículo (CENTENO, 2007). As desvantagens no seu uso referem-se ao custo elevado atrelado a operação do ROV em conjunto com a sua base de comunicação, e do cabo umbilical limitar a sua utilização em missões muito complexas. Se os cabos de comunicação e de energia utilizados quebrarem, perde-se a comunicação com o robô, e no pior dos casos se houver o rompimento total do cabo perde-se o robô no ambiente utilizado.

Os AUVs são veículos autopropulsionados que tipicamente são lançados ao mar a partir de uma embarcação, e podem atuar independentemente do navio por períodos de algumas horas à vários dias. Atualmente são muito utilizados em trajetórias lineares, como aplicações na geociência, nas quais a altitude utilizada é constante para a obtenção

de mapas e perfis do fundo do mar (WYNN et al., 2014). Também possuem diversas aplicações potenciais na área militar, monitoramento de ambiente, indústria do petróleo e inspeções subaquáticas.

Os AUVs possuem um custo operacional alto devido á logística de sua operação. Além disso, ainda existem tópicos importantes de pesquisa para tornar o veículo totalmente autônomo e confiável, como tecnologias de comunicação, sensores para navegação e fonte de energia (YUH, 2000). Para o uso em operações mais complexas, o veículo necessita de algum tipo de inteligência artificial responsável por reagir rapidamente a mudanças do ambiente, localizar locais previamente visitados através das pistas perceptivas extraídas do ambiente, bem como o processamento da movimentação do robô utilizando o mínimo de tempo e de energia.

Um dos primeiros desafios no uso dos veículos não tripulados subaquáticos está relacionado com o posicionamento georeferenciado. O *Global Positioning System* (GPS) e *Differential Global Positioning System* (DGPS) comumente utilizados em abordagens terrestres, sofrem com a atenuação das ondas eletromagnéticas no meio subaquático, tornando-se uma opção inviável. Novas abordagens são propostas, onde alia-se a posição georeferenciada através do GPS em bases na superfície em conjunto com os sensores baseados em ondas acústicas, porém existindo a limitação das missões estarem confinadas a área do alcance acústico.

A descrição do ambiente através das pistas perceptivas também é afetada em ambientes subaquáticos, uma vez que *Light Amplification by Stimulated Emission of Radiation* (LASER) e câmera óptica, habitualmente utilizados como sensores para descrever o ambiente na robótica terrestre, sofrem a interferência da turbidez da água. Nesse contexto, os sensores acústicos são menos sensíveis à turbidez da água e geralmente possuem um longo alcance, porém seus dados são de baixa resolução, mais suscetíveis a ruídos e, muitas vezes, difíceis de serem interpretados (HURTÓS; CUFI; SALVI, 2010).

Devido à inexistência de um cabo umbilical para a comunicação entre o a superfície e o AUV, torna-se necessário utilizar abordagens também baseadas em ondas acústicas para estabelecer a comunicação. Entretanto, atualmente as tecnologias existentes ocasionam perda de sinal e consequentemente atraso na aquisição da informação correta, impossibilitando o controle remoto do AUV em tempo real.

Outro fator importante a considerar é o consumo de energia, o AUV deve ser capaz de processar as informações percebidas do ambiente, a sua localização e os algoritmos de controle internamente, conduzindo a necessidade de otimização do gasto de energia utilizada pelos recursos de software e hardware. Alguns AUVs limitam o seu uso conforme o tempo de missão utilizando um conjunto de baterias (horas/poucos dias), e seguidamente o veículo retorna à superfície, enquanto outros AUVs são equipados com baterias recarregáveis podendo executar as funções por períodos maiores (dias/meses).

A navegação é outro desafio a ser tratado no ambiente subaquático. Devido às suas características são utilizados sensores predominantemente do tipo exteroceptivos, onde executa-se a fusão de informação registradas por estes sensores para estimar a localização. Também a navegação sofre com os efeitos das correntes marítimas, umas vez que os motores dos veículos precisam ter propulsão suficiente para vencê-las. Além disso, as correntezas geralmente assumem um padrão em formato de redemoinhos, com diâmetro variando de centímetros a quilômetros e podendo durar de poucos segundos até meses (PETRES et al., 2007).

Entre os diversos tópicos em relação a autonomia dos veículos subaquáticos, Localização e Mapeamento Simultâneos (SLAM) (do inglês, *Simultaneous Localization and Mapping*) é uma área-chave de estudo (DURRANT-WHYTE; BAILEY, 2006). Considerado um problema resolvido em ambientes terrestres, o SLAM ainda está em aberto no ambiente subaquático, principalmente pelas limitações sensoriais apresentadas bem como pela característica não estruturada e desconhecida do ambiente (SILVEIRA, 2015).

O problema do SLAM refere-se à possibilidade de um robô ser colocado em uma localização desconhecida em um ambiente desconhecido e o robô construir incrementalmente um mapa consistente do ambiente enquanto determina simultaneamente sua localização dentro deste mapa (DURRANT-WHYTE; BAILEY, 2006).

Embora a maior parte dos trabalhos em SLAM subaquático utilizarem métodos probabilísticos (THRUN et al., 2004; EUSTICE; PIZARRO; SINGH, 2008; BURGUERA; GONZÁLEZ; OLIVER, 2011), abordagens bioinspiradas foram propostas para resolver o problema do SLAM.

Algumas destas propostas para os algoritmos são baseados em sistemas biológicos, mais precisamente no funcionamento das células de localização e mapeamento presentes no cérebro dos roedores e seres humanos. Pesquisas envolvendo o estudo dos cérebros dos ratos, sugerem que diversas redes neurais estão envolvidas na localização e mapeamento, entre estas destacam-se as três principais células: *Place Cells* (O'KEEFE; DOSTROVSKY, 1971), *Grid Cells* (HAFTING et al., 2005) e *Head Direction Cells* (TAUBE; MULLER; RANCK, 1990).

Através dessas descobertas foram desenvolvidos algoritmos para resolução de SLAM terrestre como o RatSLAM (MILFORD; WYETH, 2008), assim como o DolphinSLAM (SILVEIRA et al., 2015; SILVEIRA, 2015) o qual foi desenvolvido pelo grupo de pesquisa NAUTEC para localização e mapeamento de robôs em ambientes subaquáticos. Ambos os algoritmos reconhecem o ambiente a partir da combinação do comportamento da percepção visual e da rede neural, o que para o ambiente subaquático torna-se uma vantagem devido à ambiguidade encontrada.

O DolphinSLAM apresentou o desempenho esperado quando utilizado em ambi-

entes simulados com imagens ópticas, entretanto em experimentos com dados coletados em ambientes reais com imagens acústicas não ocorreu o mesmo. Devido à esse comportamento, Santos (2016), Machado et al. (2015), Santos et al. (2016) apresentaram um método para realizar a detecção de ambientes semelhantes utilizando imagens acústicas.

Por outro lado, para o problema de SLAM subaquático, a presença da correnteza constitui num importante fator responsável por afetar negativamente a navegação e controle do robô e consequentemente a criação dos mapas representativos do ambiente. Desta forma, são utilizados equipamentos de sensoriamento embarcado para mensurar a correnteza, como o *Acoustic Doppler Current Profiler* (ADCP), ou na sua ausência, modelos matemáticos para estimar as correntes marítimas (HOLLINGER et al., 2012).

Nesse contexto, é proposto modificar a rede neural do algoritmo DolphinSLAM para interpretar os dados das correntes marítimas para criar um mapa mais condizente do ambiente. Para investigar o funcionamento desta modificação, o algoritmo é testado com diferentes conjuntos de dados em ambientes reais e simulados. Na próxima seção serão apresentados os objetivos e a estrutura do texto.

1.1 Objetivos

Este trabalho tem como objetivo geral estudar o problema de SLAM subaquático associado ao tratamento das correntezas marítimas a partir de modelos bioinspirados. Busca-se adaptar o algoritmo DolphinSLAM para interpretar os dados provenientes dos sensores de correnteza marítima dentro da sua rede neural. Serão apresentados resultados utilizando conjuntos de dados (*datasets*) reais e simulados.

Os objetivos específicos do trabalho são:

- Realizar uma pesquisa bibliográfica sobre o estado da arte de tecnologias de sensoriamentos utilizadas na robótica subaquática, e classificá-los entre proprioceptivos e exteroceptivos bem como as suas vantagens e desvantagens no uso.
- Realizar um estudo sobre os principais trabalhos em SLAM subaquático, elencando as principais características como o tipo de pista visual, tipo de veículo, o algoritmo e ambiente aplicado no estudo.
- Realizar uma proposta de melhoria do algoritmo DolphinSLAM (SILVEIRA, 2015) e aprimorar os módulos existentes.
- Propor uma metodologia para integrar a informação da estimação da correnteza ao modelo conexionista *Continuous Attractor Neural Network* (CANN) durante a fase de treinamento dos pesos da CANN.

- Divulgar à comunidade científica a captura dos dados para SLAM subaquático, com imagens ópticas, imagens acústicas e sensores que fornecem informações sobre o deslocamento do veículo.

1.2 Estrutura do Texto

No Capítulo 2 é apresentado o problema do SLAM e as principais abordagens utilizadas para a resolução do problema no ambiente subaquático. Também apresenta-se a classificação das abordagens em probabilísticos e bioinspirados.

No Capítulo 3 contextualizam-se os principais desafios encontrados na robótica subaquática. A partir destas características, também são apresentados e explanados o princípio de funcionamento dos sensores utilizados para a navegação. Por último, são apresentados os principais trabalhos na área do SLAM subaquático, explicando as suas principais características.

No Capítulo 4 apresenta-se o algoritmo bioinspirado para a resolução do problema de SLAM subaquático DolphinSLAM. Os módulos utilizados pelo algoritmo serão explicados e apresenta-se os equacionamentos que regem o algoritmo.

No Capítulo 5 apresenta-se o modelo de rede neural a ser implementado no algoritmo DolphinSLAM para a inclusão da informação correspondente a estimação das correntes. Os resultados são apresentados no Capítulo 6, onde utiliza-se o modelo exposto no Capítulo 5 para a interpretação do deslocamento ocasionado pela correnteza. Nesta etapa, apresenta-se primeiramente a etapa de treinamento da rede, e posteriormente o seu comportamento durante o reconhecimento da correnteza.

O Capítulo 7 apresenta as conclusões do trabalho e perspectivas de trabalhos futuros que podem ser realizados na abordagem.

2 Localização e Mapeamento Simultâneos

Durrant-Whyte e Bailey (2006) define o problema do SLAM como a possibilidade de um robô móvel ser colocado em uma localização desconhecida dentro de um ambiente desconhecido, e incrementalmente construir um mapa consistente do ambiente enquanto simultaneamente determina a sua localização neste mapa.

Atualmente existem métodos com resultados satisfatórios para ambientes pequenos, estruturados e estáticos. Entretanto, para ambientes grandes, não estruturados, com poucas pistas perceptivas disponíveis, o SLAM continua sendo um problema não resolvido.

Neste capítulo serão descritas as principais características do SLAM, apresentando a formulação clássica, diferença entre o modo completo e online, tipos de mapas gerados e as principais abordagens utilizadas para a sua resolução.

2.1 Formulação do Problema de SLAM

A Figura 1 apresenta a trajetória de um robô em um ambiente desconhecido, nela são apresentadas as diferenças entre as observações reais e as observações estimadas pelo robô para o algoritmo de SLAM. A partir dessa movimentação é criado o mapa representativo do ambiente.

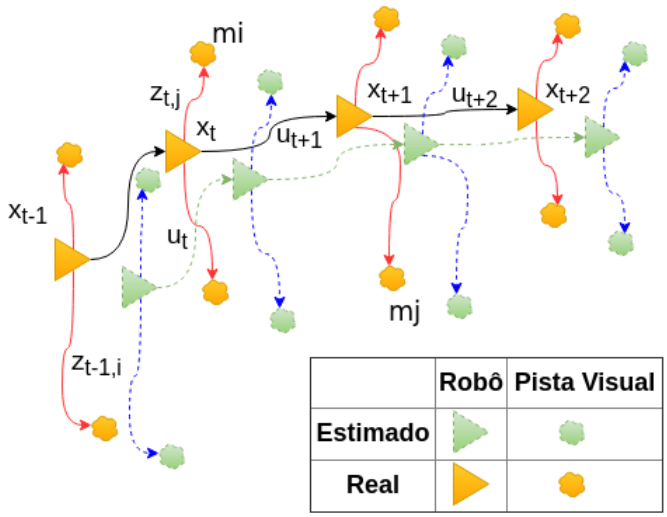


Figura 1 – O problema descrito pelo SLAM. Uma vez que as posições verdadeiras não são conhecidas, o mapa criado pelo algoritmo dependerá do robô e da localização das pistas armazenadas. Imagem baseada em (DURRANT-WHYTE; BAILEY, 2006).

A cada instante de tempo t, são definidas as seguintes variáveis para uma posição do mapa (DURRANT-WHYTE; BAILEY, 2006):

- x_t: Vetor de estados que descreve a orientação e localização do robô.
- u_t: É o vetor de controle aplicado no instante $t-1$ para locomover o veículo para o estado x_t no instante atual (t).
- m_i: Vetor responsável por armazenar a localização das pistas perceptivas do ambiente, assumindo que ela é invariante durante o tempo. Onde i refere-se à pista perceptiva visualizada.
- z_{it}: Localização relativa das pistas perceptivas em relação ao veículo.

A dimensão de representação da orientação e da localização do robô dependerá do tipo de navegação realizada. Se o ambiente for $2D$, utiliza-se um vetor de três estados, onde dois estados são responsáveis por prover a informação de localização (x, y) e um para a orientação do robô. Em ambientes $3D$, normalmente utiliza-se um vetor com seis estados, onde três referem-se a posição (x, y, z) e outros três a orientação (*roll, pitch, yaw*). A partir do total de estados armazenados durante a trajetória do veículo, é definida na Equação 2.1 o comportamento $X_{0:t}$ do robô:

$$X_{0:t} = \{x_0, x_1, \ldots, x_T\}, \tag{2.1}$$

onde, x_0 é a posição inicial do robô, e x_T a final.

O vetor de controle é responsável por representar a informação de controle enviada ao sistema de atuação do robô. Para isso integra-se os valores mensurados pela odometria a partir de sensores de orientação, aceleração, velocidade, e assim, cria-se um mapa com essa informação, chamado de *Dead-Reckoning*. Para o *ground truth* assume-se o conhecimento dos dados, e através disso, cria-se uma trajetória real do experimento. O *Dead-Reckoning* possui comportamento diferente do *ground-truth*, uma vez que os sensores possuem erros intrínsecos, e são utilizadas técnicas de integração no dados coletados pelos sensores.

Armazena-se toda a informação de controle no vetor de estados U_t (Equação 2.2):

$$U_t = \{u_1, u_2, \ldots, u_T\}, \tag{2.2}$$

onde, u_t são os sinais enviados para robô entre a posição x_{t-1} e x_t. Com os valores de X_t e U_t é possível observar/recuperar as posições anteriores do robô.

São definidos mais dois conjuntos de dados: m e Z_t. m é o mapa real do ambiente, onde são adicionadas todas as pistas perceptivas armazenadas pelo robô. Z_t refere-se a localização da pista perceptiva no mapa (m) em função da posição atual do robô (x_t) conforme demonstrado na Equação 2.3:

$$Z_t = \{z_1, z_2, \ldots, z_T\}. \tag{2.3}$$

A partir das definições mostradas nessa seção, é possível afirmar que o problema do SLAM está relacionado com recuperar um modelo do mapa (m) e das localizações (X_t), utilizando o vetor de controle (U_t) e as pistas perceptíveis (Z_t). Nesse contexto, são utilizadas duas notações para o SLAM: offline e online (THRUN; LEONARD, 2008).

2.1.1 SLAM Offline

No SLAM completo ou offline, primeiramente é realizada toda a trajetória de movimento do robô enquanto são armazenados os dados. Após isso, é criado o mapa do caminho do robô levando-se em conta a odometria, pistas perceptíveis e a probabilidade do robô ter passado naquele ponto, como demonstrado na Equação 2.4:

$$P(X_T, m|Z_T, U_T). \tag{2.4}$$

Escrita dessa forma, essa abordagem baseia-se em calcular a probabilidade posterior conjunta sobre X_T e m com base em todos os dados adquiridos até o momento atual (SILVEIRA, 2015). As variáveis presentes à direita da barra condicional (Z_T,U_T) são observáveis, onde U_T é o sinal de controle e Z_T são observados através dos sensores do robô, enquanto que as variáveis à esquerda (X_T,m) são as saídas do algoritmo.

Outra característica do SLAM offline, é o processamento de uma grande quantidade de dados ao mesmo tempo, assim aumentando o custo computacional da aplicação. Também, o uso dessa abordagem é limitada, visto que a ideia é utilizar o robô para tarefas autônomas.

2.1.2 SLAM Online

O SLAM online possui a característica de cálculo incremental da localização atual do robô, em oposição ao caminho inteiro como utilizado no SLAM offline. O problema é representado pela Equação 2.5:

$$P(x_t, m|Z_T, U_T). \tag{2.5}$$

Devido à essa característica, os algoritmos online são geralmente incrementais e processam os dados a cada instante de tempo.

Para ambas as abordagens é desejável o uso de soluções recursivas para o problema de SLAM, assim, é utilizado o Teorema de Bayes. Para tal, define-se o modelo de movimento e o modelo de observação (DURRANT-WHYTE; BAILEY, 2006):

- Modelo de movimento: É responsável por descrever a probabilidade do movimento do robô entre dois estados, conforme demonstrado na Equação 2.6:

$$P(x_t|x_{t-1}, u_t). \tag{2.6}$$

A transição entre os dois estados é baseado na propriedade de *Markov*, onde a posição atual (x_t) depende somente da posição do instante anterior (x_{t-1}) mais o sinal de controle do robô (u_t).

- Modelo de observação: Nessa etapa é calculada a probabilidade de realizar a observação de uma pista perceptiva quando a localização do veículo e da pista perceptiva do mapa são conhecidas, conforme a Equação 2.7:

$$P(z_t|x_t, m). \tag{2.7}$$

Com essas definições torna-se possível implementar um algoritmo probabilístico de SLAM.

2.2 Tipos de Mapa

O mapa criado pelo algoritmo de SLAM é responsável por descrever o seu ambiente a partir da trajetória do robô. Existem alguns autores que utilizam a classificação dos mapas em: mapas métricos, mapas topológicos, mapas de características e mapas probabilísticos (ENGEL, 2003; SELVATICI, 2009).

Porém, nesse trabalho será utilizada a abordagem utilizada por Thrun e Leonard (2008), onde são definidos dois tipos de mapas criados por um sistema de SLAM: Os mapas topológicos e os mapas métricos.

- Mapas Métricos: São os mapas que representam o ambiente com as informações métricas entre os locais do mapa. Por exemplo o ponto x_1 está a 5 metros do ponto x_2.

- Mapas Topológicos: Os mapas topológicos fornecem uma informação qualitativa relativa entre os lugares do mapa. Nas últimas décadas, os estudos na área da neurociência sugerem que os mamíferos como seres humanos e ratos utilizam as informações topológicas para navegação. O princípio de funcionamento do mapa pode ser escrito como o ponto x_1 está a cima do ponto x_2.

2.3 Abordagens para Resolução do SLAM

Durante as últimas décadas, diversas abordagens foram utilizadas para a resolução do problema do SLAM, e consequentemente produzindo inúmeras pesquisas na área. As abordagens probabilísticas tornaram-se amplamente utilizadas, onde os principais algoritmos são baseados no *Extended Kalman Filter* (EKF) (MALLIOS et al., 2010; FERREIRA et al., 2012; DISSANAYAKE et al., 2001b) e o Filtro de Partícula comumente chamado na literatura de FastSLAM (THRUN, 2001),(KOUZOUBOV; AUSTIN, 2004). Ambas as soluções necessitam dos modelos do robô (Equação 2.6), e do modelo das pistas perceptivas (Equação 2.7).

Outra linha de pesquisa tornou-se promissora a partir do avanço nas pesquisas dos neurônios de localização nos ratos. Utilizando os modelos descobertos de localização e mapeamento, são desenvolvidos algoritmos bioinspirados para resolução do problema de SLAM, onde o RatSLAM (MILFORD; WYETH, 2008) encontrou resultados promissores para SLAM terrestre. Em 2015, (SILVEIRA, 2015; SILVEIRA et al., 2015) apresentou o DolphinSLAM, consiste numa extensão do RatSLAM para ambientes subaquático explicado com mais de detalhes no Capítulo 5.

Nas próximas subseções serão apresentados os principais algoritmos probabilísticos e bioinspirados.

2.3.1 Abordagens Probabilísticas

O SLAM probabilístico congrega o maior quantitativo de abordagens presentes na literatura. Foram desenvolvidas diversas abordagens utilizando como base os algoritmos do EKF e do Filtro de Partículas, desta forma, nessa seção o princípio de funcionamento dos dois algoritmos serão apresentados.

2.3.1.1 Filtro de Kalman Estendido

Historicamente, o uso do EKF para SLAM é considerada a primeira abordagem proposta, e assim, tornando-se o mais antigo e ainda o mais utilizado nos dias atuais. É uma versão estendida do Filtro de Kalman (KALMAN, 1960) original.

Em 1986 (SMITH; CHEESEMAN, 1986), foi apresentado o trabalho pioneiro com EKF, utilizando um único vetor de estados para estabelecer a localização do robô e o conjunto de pistas perceptivas do ambiente. Para isso, foram associados os erros da matriz de covariância representando a incerteza das estimações, incluindo as relações entre o veículo e a estimativa da pista perceptiva (THRUN; LEONARD, 2008). O mapa gerado pelo algoritmo é métrico, onde o ambiente é representado em conjunto com as pistas perceptivas.

O algoritmo EKF pode ser descrito a partir de uma distribuição gaussiana multivariada conforme demonstrada na Equação 2.8:

$$p(x_t, m|Z_T, U_T) \sim N(\mu_t, \sum t), \tag{2.8}$$

onde, $\sum t$ refere-se a matriz de covariância em relação a μ_t, μ_t fornece a localização das pistas perceptivas e a melhor estimativa da posição do robô no ambiente.

Durrant-Whyte e Bailey (2006) descreve um modelo aproximado do modelo de movimentação e de observação através da distribuição Gaussiana multivariada.

O modelo de movimentação representa a estimativa da posição do robô (x_t), calculada através da posição anterior (x_{t-1}) em conjunto com a ação de controle (u_t) as quais são aplicadas na função $g(x_{t-1}, u_t)$. R_t acrescenta os distúrbios na matriz de covariância, como descrito na Equação 2.9:

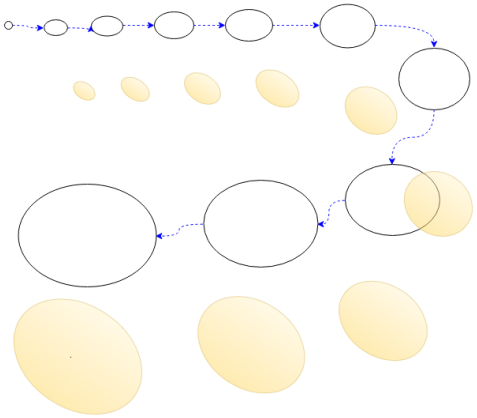
$$p(x_t|x_{t-1}, u_t) \sim N(g(x_{t-1}, u_t), R_t). \tag{2.9}$$

No modelo de observação são ponderados os possíveis lugares das pistas perceptivas. Assim, utiliza-se ($h(x_t, m)$) para retornar a localização das pistas visuais, e Q_t é a matriz que representa o erro conforme demonstrado na Equação 2.10:

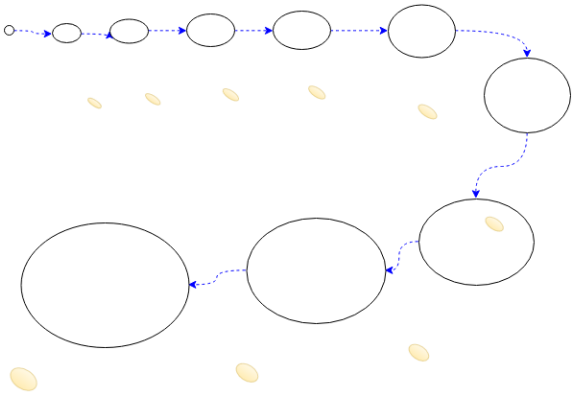
$$p(z_t|x_t, m) \sim N(h(x_t, m), Q_t). \tag{2.10}$$

Na Figura 2 é apresentado o funcionamento do EKF. A Figura 2(a) mostra a trajetória realizada pelo robô em um ambiente qualquer, onde as elipses sem preenchimento referem-se a estimativa do movimento do robô, e as elipses preenchidas por amarelo referem-se a estimativa das posições das pistas visuais do ambiente. Na Figura 2(b), o robô reconhece uma pista perceptiva anteriormente visualizada, assim corrigindo a estimativa de posição do robô, e consequentemente diminuindo a incerteza das pistas visuais armazenadas.

As principais desvantagem no uso do EKF é o esforço computacional crescente a cada nova pista perceptiva adicionada ao vetor de estados do algoritmo, e a matriz de covariância que possui o crescimento quadrático. Nesse contexto, o algoritmo necessita de alguma métrica para limitar o número de pistas visuais ou utilizá-lo em pequenos ambientes. Bailey e Durrant-Whyte (2006) apresentam algumas variações do EKF para diminuir o custo computacional.



(a) Comportamento do algoritmo durante a trajetória.



(b) Comportamento do algoritmo durante a trajetória após o reconhecimento de alguma pista previamente visualizada no ambiente.

Figura 2 – Exemplo do algoritmo do EKF para o problema do SLAM. Imagem baseada em (THRUN; LEONARD, 2008).

2.3.1.2 Filtro de Partículas

O filtro de partículas é uma técnica sequencial de Monte Carlo, fundamentado pela estatística Bayesiana, onde são realizados cálculos das distribuições de probabilidades relevantes através dos conceitos de amostragem. Através das distribuições são aproximadas as medidas aleatórias discretas, as quais são designadas como partículas com pesos específicos (DJURIC et al., 2003).

Esta metodologia encontrava-se limitada a algumas aplicações, devido à sua complexidade computacional relativamente elevada (RIBERIO, 2012). Nos últimos anos, estas pesquisas voltaram a ser atrativas a partir dos avanços na área de processamento computacional.

Nesse contexto, Montemerlo et al. (2002a) apresentaram o algoritmo FastSLAM, o qual mostrou resultados promissores na resolução do problema do SLAM. O FastSLAM utiliza o filtro de partículas para a estimativa do caminho percorrido pelo robô. Segundo Thrun e Leonard (2008), a cada iteração, o FastSLAM armazena K partículas, conforme demonstrado na Equação 2.11:

$$X_t^{[k]}, \mu_{t,1}^{[k]}, \ldots, \mu_{t,N}^{[k]}, \sum_{t,1}^{[k]}, \ldots, \sum_{t,N}^{[k]}, \tag{2.11}$$

onde $[k]$ é o número de amostras, e cada partícula do sistema é composta por:

- $X_t^{[k]}$**:** A trajetória estimada do robô.
- N: Um conjunto de gaussianas 2D, com a média $(\mu_{t,N}^{[k]})$ e a variância $\sum_{t,N}^{[k]}$, para cada pista visual do ambiente.

A partir disso, as K partículas estimam K possíveis caminhos. Também possui KN Gaussianas, onde cada um dos modelos representa exatamente uma pista perceptiva para uma partícula(THRUN; LEONARD, 2008).

Para a utilização do filtro tornar-se viável, o FastSLAM utiliza três técnicas: *Rao-Blackwellization*, independência condicional e re-amostragem. Na Figura 3 apresenta-se um exemplo de trajetória criada pelo FastSLAM.

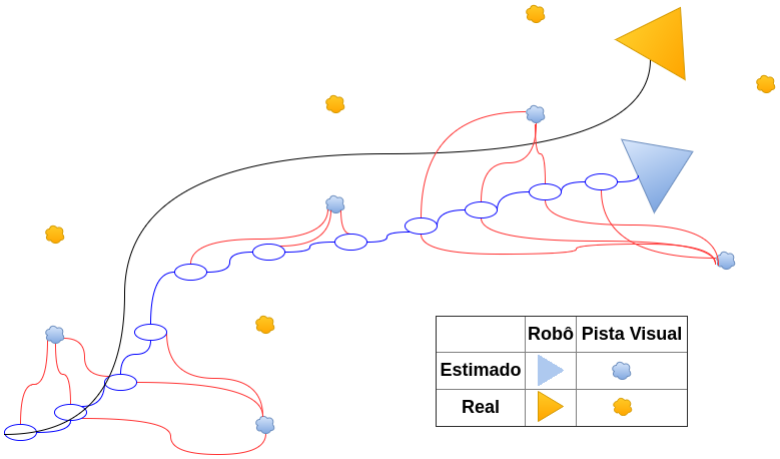


Figura 3 – Exemplo de Trajetória do FastSLAM.

A trajetória real do robô é definida pela linha preta e o robô pelo triângulo amarelo. Na estimativa as elipses são relacionadas a cada estágio de atualização e as linhas vermelhas referem-se a pista perceptiva visualizada pela partícula.

2.3.1.3 Vantagens e Desvantagens dos Métodos

As duas propostas apresentadas na seção anterior representam a predominância dos trabalhos na área do SLAM. Cada uma das abordagens possui suas vantagens e desvantagens, devendo ser ponderados na escolha.

O EKF SLAM possui um alto custo computacional devido à atualização quadrática das pistas visuais do ambiente, sendo necessário utilizar algumas limitações. Algumas soluções são que permitem mitigar o custo computacional são discutidas em Bailey e Durrant-Whyte (2006). Também Thrun e Leonard (2008) destacam que as abordagens que utilizam submapas produzem resultados semelhantes a abordagens baseadas em grafos, utilizando menor tempo de atualização.

O FastSLAM possui a vantagem de ser computacionalmente mais eficiente que o EKF, uma vez que a sua atualização possui custo linear-logarítmico. Entretanto, o FastSLAM possui a desvantagem do número de partículas poder ser extremamente grande quando utilizado em ambientes com grandes números de reconhecimento de lugares previamente visitados. Porém, alguns trabalhos (FOX; BURGARD; THRUN, 2003), (YO-

KOZUKA; MATSUMOTO, 2012), (CAIN; LEONESSA, 2016) utilizam a forma de *grid* para diminuir o número de partículas empregadas e o custo computacional envolvido.

2.3.2 Abordagens Bioinspiradas

Em contrapartida da grande maioria de abordagens para resolver o problema do SLAM ser probabilísticas, surgiram novas abordagens, entre elas, as abordagens bioinspiradas.

Os algoritmos bioinspirados como o nome sugere, são sistemas baseados em seres vivos, como por exemplo os roedores, os quais possuem um sistema de localização capaz de navegar e se posicionar em relação as pistas visuais armazenadas do ambiente no qual está inserido. Nesse contexto, Nafouki e Conradt (2015) definem que os sistemas de SLAM bioinspirados utilizam tanto a odometria como as pistas visuais adquiridas por sistemas visuais para atualizar e corrigir a estimativa de posição do robô e o mapa gerado. Outra vantagem do uso das abordagens bioinspiradas refere-se a ausência do modelo matemático do veículo utilizado e das múltiplas hipóteses geradas a partir dos métodos probabilísticos.

As abordagens bioinspiradas para SLAM que serão abordadas nessa seção surgiram através de estudos na região do Hippocampo do cérebro dos roedores, onde foram encontradas células com características específicas para localização e mapeamento do ambiente: *Place Cells*, *Head Direction Cells* e *Grid Cells*, as quais serão descritas a seguir.

2.3.2.1 Place Cells

No começo dos anos 70, as *Place Cells* foram descobertas por O'Keefe e Dostrovsky (1971) enquanto realizavam experimentos na parte dorsal do Hippocampo de um roedor que executava movimentos livres no ambiente. Os neurônios encontrados possuíam um padrão de disparo relativo ao local onde o animal encontrava-se no ambiente. Cada *Place Cell* é responsável por descrever uma porção do ambiente através da informação sensorial coletada pelo animal e possuindo um padrão de disparo único, e assim, criando mapas cognitivos que representam as pistas perceptivas no ambiente.

As mesmas *Place Cells* ativas em um ambiente, podem estar ativas em outros ambientes, mas possuindo um comportamento de disparo totalmente diferente (O'KEEFE; NADEL, 1978). Foi observado que o Hippocampo dos roedores possuem múltiplos mapas, definidos a partir da combinação das *Place Cells* ativas.

Outra característica deste tipo de célula diz respeito à manutenção dos campos de disparo na ausência de informações sensoriais externas. Quando o rato está navegando em um ambiente e a luz é apagada, os campos de disparo das *Place Cells* são mantidos por alguns minutos, implicando que o mamífero utiliza informações de movimentação própria como entrada nos neurônios. Nesse contexto, Nafouki e Conradt (2015) afirmam que cada

Place Cell recebe a entrada a partir das pistas perceptivas do ambiente e de estímulos internos do seu sistema de navegação. Na Figura 4 é apresentado um exemplo de ativação dos neurônios do tipo *Place Cell*.

Figura 4 – Exemplo de Ativação de *Place Cell*.

2.3.2.2 Head Direction Cells

Em 1990, Taube, Muller e Ranck (1990) apresentaram uma classe de neurônios localizada em camadas profundas do *Postubiculum*, as quais apresentaram uma taxa de disparo precisa em relação a orientação da cabeça do rato. Esse neurônio recebeu o nome de *Head Direction Cells*.

As *Head Direction Cells* também são ativadas a partir das pistas perceptivas. Quando as informações são avaliadas em conjunto com as *Place Cells* fornecem um sistema complementar, onde se tem informação de posição e orientação. Outra característica das *Head Direction Cells* é que a taxa de disparo da orientação encontrada é indiferente a posição do corpo do rato no ambiente. Na Figura 5 é apresentado o comportamento do neurônio *Head Direction*.

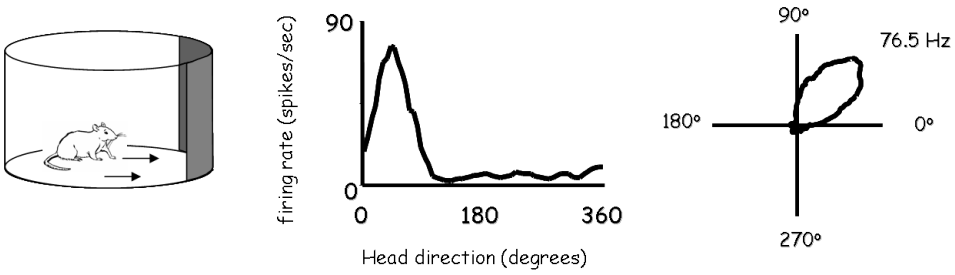


Figura 5 – Exemplo de Ativação de *Head Direction Cell*.

Fonte: <http://www.memoryspace.mvm.ed.ac.uk/headdirectioncells.html>

2.3.2.3 Grid Cells

Em 2004, Fyhn et al. (2004) descobriram a presença no Cortex Entorhinal de células onde os padrões de disparo assemelhavam-se as das *Place Cells*, entretanto com o comportamento em múltiplos disparos. Posteriormente em 2005, Hafting et al. (2005) realizaram o experimento utilizando o animal em ambientes maiores, onde descobriu as *Grid Cells* com um formato de disparo triangular. Também, outra características é que o mapeamento é realizado em forma de *grid*, indicando a possibilidade dos animais utilizarem um sistema métrico para navegação.

Cada *grid* é caracterizado pelo espaçamento dos disparos, orientação e fase (deslocamento) em relação a algum ponto (MOSER; KROPFF; MOSER, 2008). A *Grid Cell* cobre todo o ambiente explorado pelo animal e a sua ativação ocorre quando o animal estiver localizado em qualquer um dos vértices do padrão triangular. Embora na literatura alguns autores considerem que as *Grid Cells* apresentam um padrão de disparo em formato hexagonal. Na Figura 6 é apresentado o padrão de ativação de uma *Grid Cell*.

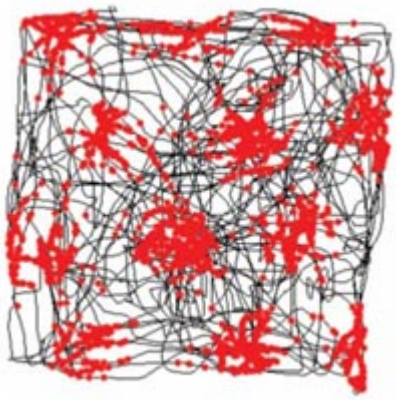


Figura 6 – Exemplo de Ativação de uma *Grid Cell*.

Fonte: <http://neurotopographics.psychol.ucl.ac.uk/science/science.html>

2.3.2.4 RatSLAM

Com o avanço nas pesquisas sobre os neurônios utilizados pelos animais para realizar a localização e mapeamento, Milford, Wyeth e Prasser (2004) observaram que as *Place Cells, Head Direction Cells e Grid cell* possuíam características que poderiam ser empregadas para resolução do problema de SLAM, e assim, propuseram um algoritmo bioinspirado. O algoritmo recebeu o nome de RatSLAM devido a sua concepção ser baseada nos neurônios descobertos nos ratos. Os resultados dos experimentos para *datasets* de curta e longa duração foram promissores, assim o algoritmo conseguiu criar um mapa consistente durante toda a navegação em tempo real. Ball et al. (2013) afirmam que o

sistema do RatSLAM pode ser dividido em três módulos principais (ver Figura 7): *Pose Cells*, *Local View Cells* e Mapa de Experiência.

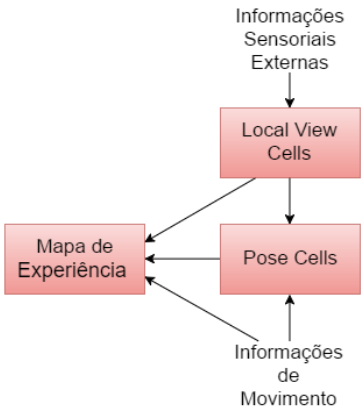


Figura 7 – Arquitetura do algoritmo RatSLAM.

A *Pose Cells* é considerada o núcleo principal do sistema RatSLAM e representa os três graus de liberdade referentes a posição do robô representado mediante uma rede CANN para um espaço de $3D$ (MILFORD; WYETH, 2008). Os três graus de liberdade mencionados, referem-se a distribuição absoluta da posição no plano $2D$ de uma *Place Cell* em conjunto com o valor θ referente a *Head Direction Cell*. Todas as faces da estrutura 3D estão conectadas com as faces opostas conforme mostrado na Figura 8:

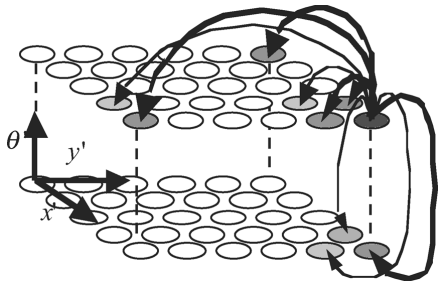


Figura 8 – CANN Utilizada pelo Algoritmo RatSLAM. Imagem extraída de (MILFORD; WYETH; PRASSER, 2004)

A atualização da *Pose Cells* é realizada através das informações de movimento do robô e a calibração da correção do mapa é realizada através da *Local View*. O mapa de experiência é responsável por receber os dados de movimento, *pose cell* e *local view cells* e prover um mapa representativo do ambiente. A seguir será apresentada a modelagem matemática do algoritmo RatSLAM e as equações utilizadas para o algoritmo RatSLAM.

Pose Cells

- Dinâmica da Rede: Na dinâmica da rede, primeiramente é criada a matriz de pesos excitatória (ε) baseada em uma distribuição gaussiana conforme demonstrada na

Equação 2.12:

$$\varepsilon_r = e^{-\frac{d^2_{x'}+d^2_{y'}}{2k_p}} e^{-\frac{d^2_{\theta'}}{2k_d}}, \tag{2.12}$$

onde, $d_{x'}$ e $d_{y'}$ refere-se a distância entre os dois neurônios, k_p e k_d são constantes de tamanho para a orientação e lugar. Todos os neurônios na rede estão conectados e propagam a excitação conforme a Equação 2.13:

$$rr_{x',y',\theta'} = \sum_{a=1}^{n_{x'}} \sum_{b=1}^{n_{y'}} \sum_{c=1}^{n_{\theta'}} \varepsilon . r_{abc}, \tag{2.13}$$

onde $rr_{x',y',\theta'}$ é a excitação entre os neurônios e $n_{x'}$, $n_{y'}$ e $n_{\theta'}$ referem-se a dimensão da rede neural. E as distâncias entre os dois neurônios atuais é calculado conforme a Equação 2.14:

$$\begin{aligned} a &= \min(|x' - a|, n_{x'} - |x' - a|), \\ b &= \min(|y' - b|, n_{y'} - |y' - b|), \\ c &= \min(|\theta' - c|, n_{\theta'} - |\theta' - c|). \end{aligned} \tag{2.14}$$

Cada célula também inibe as células vizinhas utilizando a forma inibitória da matriz de peso excitatória (MILFORD; WYETH, 2008). É proposto a inibição após a excitação para encontrar uma dinâmica de rede adequada, sem precisar utilizar uma função de distribuição do tipo Chapéu Mexicano (KOHONEN, 1995). Esta proposta é demonstrada na Equação 2.15

$$rr_{x',y',\theta'} = \sum_{a=1}^{n_{x'}} \sum_{b=1}^{n_{y'}} \sum_{c=1}^{n_{\theta'}} \omega . r_{abc} - \phi, \tag{2.15}$$

onde ϕ é a constante global de inibição.

Após essa etapa, é realizada a integração de caminho da rede, a partir da qual o pacote de energia é deslocado conforme os dados fornecidos pelas atualizações da odometria.

- **Local View Cells:** Por último, na etapa da *Pose Cells* é realizada a calibração a partir dos dados da *Local View*. Para isso, é realizada uma etapa de aprendizado de associação entre a *Local View* e *Pose cell*. Uma *Local View* é considerada ativa se ela for similar a alguma *Local View* armazenada anteriormente, se isto ocorrer, são reforçadas as ligações entre a *Local View* e as *Place Cells* ativas conforme a taxa correspondente, conforme demonstrado na Equação 2.16:

$$\beta r_{ix'y'\theta'} = max(\beta_{ix'y'\theta'}, \lambda l_i r_{x'y'\theta'}), \tag{2.16}$$

onde βr refere-se a matriz armazenada de conexões,λ é a taxa de aprendizado e l_i a *local view*.

Mapa de Experiência

O mapa de experiência é responsável por fornecer uma representação topológica do ambiente visitado. Para isso, são criadas experiências (e) no mapa, as quais possuem informação sobre a *pose cell* e *local view cell* referente ao momento da aquisição, e a posição no mapa. Por definição uma experiência é uma tupla, conforme descrito na Equação 2.17:

$$er_i = \{R^i, L^i, p^i\}, \tag{2.17}$$

onde, er_i é a experiência atual, R^i a ativação instantânea da rede, L^i a *local view* ativa, e p^i a posição do robô no mapa. A primeira experiência do mapa é criada em um ponto qualquer do mapa, porém as subsequentes dependerão da primeira mais o valor da transição proveniente da odometria.

Uma nova experiência, somente é criada se nenhuma outra experiência já visitada possuir taxa de similaridade mais alta que um limiar pré-definido. A Equação 2.18 é utilizada para calcular a taxa para cada possível nova experiência:

$$S_r = \alpha_p |R^i - R| + \alpha_l |L^i - L|, \tag{2.18}$$

onde S_r é a taxa calculada, α_p e α_l são constantes de peso da importância da rede neural ou da *local view*. Quando uma nova experiência é criada, calcula-se a transição entre a posição atual e a anterior através da odometria, conforme a Equação 2.19:

$$tr_{ij} = \{\Delta p^{ij}\}. \tag{2.19}$$

Com o valor da transição calculado, uma nova experiência é criada como demonstrado na Equação 2.20:

$$er_j = \{R^j, L^j, p^i + \Delta p^{ij}\}. \tag{2.20}$$

A última etapa do algoritmo é a correção do mapa. No RatSLAM não existe a detecção explicita de *loop*. O fechamento de *loop* ocorre quando a experiência atual possuir similaridade com alguma experiência anterior, e assim o erro entre as duas posições é propagado por todo o sistema utilizando a Equação 2.21:

$$\Delta pr^i = \alpha \left[\sum_{j=1}^{N_f} (p^j - p^i - \Delta p^{ij}) + \sum_{k=1}^{N_t} (p^k - p^i - \Delta p^{ki}) \right], \tag{2.21}$$

onde α é a constante de correção, p_i é a posição atual do robô, Δp^{ij} é a distância percorrida entre a posição p_i e p_j, N_f o número experiências e_i indo para outras experiências, e N_t é o número de experiências chegando em e_i.

Outra característica do RatSLAM é que a correção do mapa é realizada a cada nova iteração do algoritmo, porém, enquanto não houver reconhecimento de locais semelhantes, a Equação 2.21 possui comportamento nulo. Após o reconhecimento, ocorre as correções do mapa, as quais são executadas por todas as experiências.

Neste capítulo apresentou-se a formulação do problema do SLAM, e as diferenças entre os métodos online e offline. Também apresentou-se o EKF e FastSLAM, métodos probabilísticos mais utilizados para a resolução do SLAM. Como alternativa a esses métodos, apresentou-se o modelo bioinspirado RatSLAM, é baseado nas *Head Direction Cells*, *Place Cells* e *Grid Cells*. No Capítulo 3 serão apresentados os principais desafios para o SLAM subaquático, os sensores utilizados e os principais trabalhos para a sua resolução.

3 SLAM Subaquático

Neste capítulo serão apresentados os desafios encontrados nos meios subaquáticos, posteriormente são apresentados e explanados os sensores utilizados, e por último são apresentados os principais artigos associados ao SLAM subaquático. No fim do capítulo serão apresentados na Tabela 1 o robô, tipo do ambiente, sensores e o algoritmo utilizado em cada abordagem.

3.1 Desafios do Meio Subaquático

Em comparação com o SLAM terrestre, o SLAM subaquático possui dificuldades adicionais como as correntes marítimas, alta pressão, ambiente desestruturado e desconhecido, dificuldade na extração de *features*, limitação do alcance e da precisão dos sensores e a impossibilidade do uso de sinais geo-referenciados.

Sensores comumente utilizados em ambientes terrestres como câmera e *Sound Navigation and Ranging* (SONAR), possuem restrições no ambiente subaquático devido ao padrão de propagação e espalhamento da luz, e extração de *features*. Na Figura 9 (CODEVILLA et al., 2014) é apresentada a dificuldade para extrair *features* significativas nas imagens ópticas devido ao nível de turbidez (DREWS et al., 2013; DREWS-JR et al., 2016).

Figura 9 – Exemplo de imagem adquirida em um ambiente subaquático, onde a água afeta o transporte da luz e degrada a qualidade da imagem. Imagem extraída de (CODEVILLA et al., 2014)

A localização no ambiente subaquático é realizada diferentemente do ambiente terrestre, uma vez que o sinal eletromagnético dos satélites associados ao GPS são fortemente atenuados na água. Assim, uma das alternativas utilizadas é o uso de sensores baseados em ondas acústicas, as quais possuem grande alcance e baixa atenuação. Nesse contexto, foram desenvolvidos sensores específicos e algoritmos para o tratamento dos dados para o ambiente aquático.

3.2 Sensores Utilizados

Nesse trabalho será utilizada a classificação dos sensores como proprioceptivo e exteroceptivo.

3.2.1 Sensores Proprioceptivos

Os sensores proprioceptivos são responsáveis por mensurar os estados internos do sistema (robô). Como exemplo pode-se citar: giroscópios, potenciômetros, *encoder* ópticos e magnéticos, entre outros.

3.2.1.1 Giroscópio

O giroscópio é um sensor que retorna uma medida absoluta da direção x, y e z do robô em relação ao ponto fixo. No mercado comercial é possível encontrar diferentes tipos de giroscópios: Rotacional (DUNN, 1995), vibratório (DUNN; ROOP, 1994) e óptico (LEFEVRE, 2014).

O giroscópio rotacional consiste em um disco rotor montado em uma estrutura que se tem 3 graus de liberdade rotacionais, ou seja, o disco é livre para girar ao redor de qualquer um de seus eixos de rotação (GABRIEL, 2014). Devido aos materiais utilizados em sua composição, o sensor sofre com o acréscimo de erro durante o uso.

O giroscópio vibratório utiliza a vibração da estrutura onde está acoplado para determinar a sua rotação. Nesse contexto, foram desenvolvidos giroscópios utilizando sistemas microeletromecânicos encontrados em aplicações comerciais, como celulares, automóveis, vídeo games, entre outros.

O giroscópio óptico é baseado no efeito de Sagnac (POST, 1967), onde aplica-se em uma fibra ótica o mesmo emissor de luz em sentido horário e anti-horário e mede-se a diferença de fase na saída das duas entradas. Essa diferença de fase será proporcional a velocidade angular. A principal vantagem do sensor é não utilizar partes móveis, tornando-se assim menos suscetível aos erros encontrados nos outros tipos de giroscópios e ideal para ambientes subaquáticos.

3.2.1.2 Unidade de Medição Inercial

Uma *Inertial Measurement Unit* (IMU) típica (Figura 10), consiste em três acelerômetros e três giroscópios montados em conjunto sobre três eixos ortogonais (DISSANAYAKE et al., 2001a), nos quais os acelerômetros mensuram a aceleração do robô, e os giroscópios a orientação em relação a algum referencial. Com esta composição, é possível calcular a posição relativa, velocidade, aceleração e orientação do robô, como mostrada na Figura 11.

Figura 10 – Exemplo de um modelo de IMU Comercial

Fonte: <http://www.controlsystemslab.gr/index/images/equipment/underwater/mti2.png>

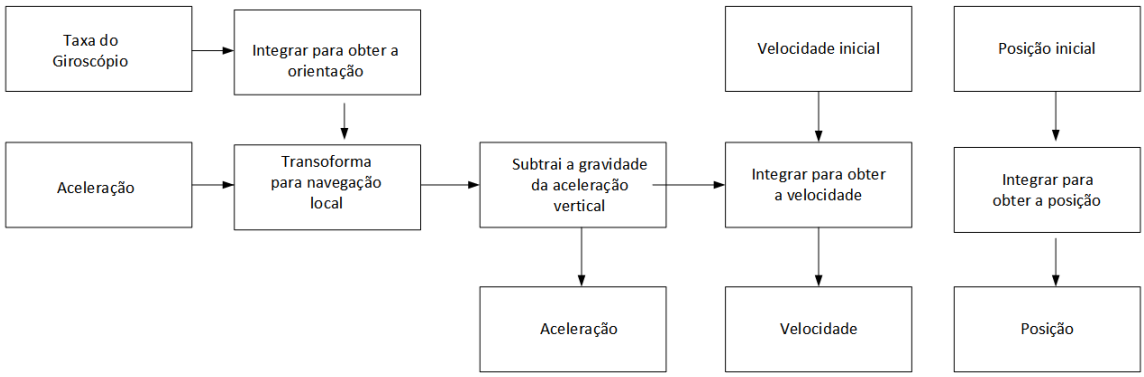


Figura 11 – Descrição do Processo de Obtenção da Aceleração, Velocidade e Posição da IMU

Fonte: <https://www.ethz.ch/content/dam/ethz/special-interest/mavt/robotics-n-intelligent-systems/asl-dam/documents/lectures/autonomous_mobile_robots/spring-2015/04_-_Perception_I_-_Sensors.pdf>

Devido à sua composição, as IMUs apresentam alta sensibilidade ao acréscimo de erros nas suas medições, tornando-se necessário utilizar alguma abordagem para cancelar o erro após algum tempo de execução.

3.2.2 Sensores Exteroceptivos

Os sensores exteroceptivos descrevem as informações relativas ao ambiente externo do robô. Diversos sensores podem ser classificados, tal como, bússola, GPS, câmera,

LASER, entre outros.

3.2.2.1 Altímetro

O altímetro (*Echo-sounders*) é responsável por fornecer a distância do veículo em relação ao fundo do mar. O sensor transmite um pulso de onda acústica direcionada ao fundo do mar e calcula o tempo para que o mesmo retorne a fonte emissora, com o tempo e a velocidade da onda, é calculada a distância. Na Figura 12, é apresentado um exemplo de altímetro comercial.

Figura 12 – Exemplo de Altímetro comercial.

Fonte: <http://www.tritech.co.uk/media/products/ede1f51bc375f825bcc01d275abcd9e4_43781.jpeg>

3.2.2.2 Sensor de Pressão

O sensor de pressão fornece qual a profundidade do robô em relação a superfície. Conforme o robô submerge na água, a pressão exercida sobre ele aumenta, e assim torna-se possível criar uma relação entre a diferença de pressão e a profundidade da água. Na Figura 13, é mostrado um exemplo de sensor de pressão utilizado em experimentos.

Figura 13 – Exemplo de Sensor de Pressão.

Fonte: <http://www.valeport.co.uk/Portals/0/Images/Valeport_miniIPS.jpg>

3.2.2.3 Doppler Velocity Log

Por mais de uma década, *Doppler Velocity Log* (DVL) tem sido utilizada para prover informação de navegação para navios, ROVs and AUVs (SNYDER, 2010). A DVL (Figura 14) utiliza múltiplos transdutores para medir a velocidade utilizando como referência o fundo ou a coluna de água. O princípio de funcionamento baseia-se na propagação das ondas, onde primeiramente é emitido ondas em diferentes direções e com o tempo de

retorno é calculado o deslocamento do robô. Quando comparado com a IMU possui maior precisão, porém também sofre com o erro acumulativo.

Figura 14 – Exemplo de DVL

Fonte: <http://rts.as/wp-content/uploads/2015/03/Teledyne-RDI-Workhorse-Navigator-Doppler-Velocity-Log.jpg>

3.2.2.4 Acoustic Doppler Current Profiler

O ADCP é o sensor utilizado para medir a estimativa da velocidade da correnteza. O seu princípio de funcionamento é similar a DVL, onde envia-se ondas sonoras em uma frequência constante na água e através do tempo em que ocorre a reflexão ocasionada pela colisão em partículas suspensas na água, calcula-se a velocidade da correnteza. Na Figura 15, apresenta-se um exemplo de ADCP:

Figura 15 – Exemplo de ADCP

Fonte: <http://seatronics-group.com/files/cache/bf3480b2e2bfd58ab7595bd64b75c389_f563.jpg>

3.2.2.5 Bússola

A bússola é considerado um dos sensores mais antigos utilizado pelo ser humano e possui a finalidade de medir a orientação do robô em função do campo magnético terrestre. A principal desvantagem da bússola é ser passível a interferência de campos magnéticos oriundos de outros equipamentos ou do ambiente. Na Figura 16 é apresentado um exemplo de bússola.

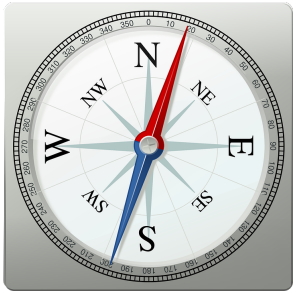


Figura 16 – Exemplo de bússola

Fonte: <https://pixabay.com/static/uploads/photo/2013/07/12/17/38/compass-152124_960_720.png>

3.2.2.6 Short Baseline Positioning System

Short Baseline Positioning System (SBL) é um método que utiliza ondas acústicas para calcular a posição do robô no meio subaquático. Nessa abordagem são utilizados no mínimo três *transceivers* espaçados entre $10-50$ metros, onde são distribuídos no casco do navio para realizar a triangulação dos sinais. Para descobrir a posição do robô é utilizado o tempo de resposta entre cada *transceiver* e a onde emitida pelo alvo. Essa configuração é mostrada na Figura 17.

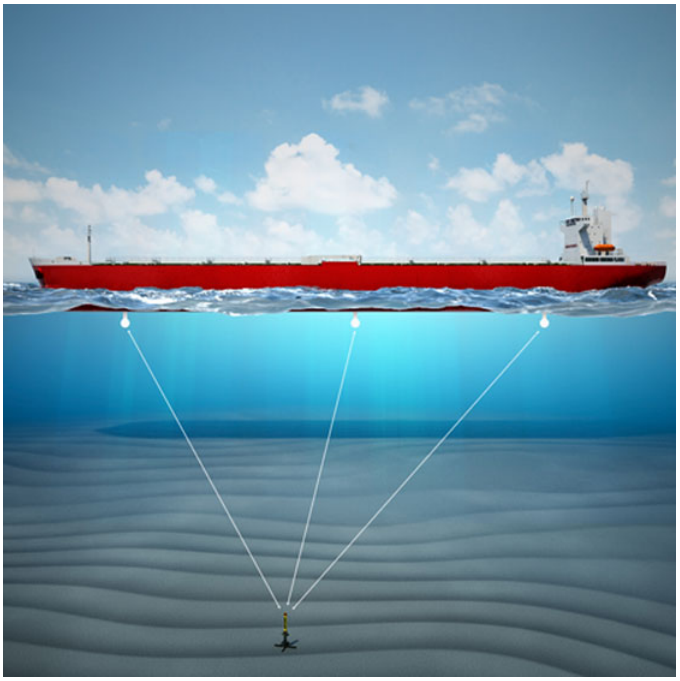


Figura 17 – Exemplo de como é realizada a comunicação em um sistema SBL

Fonte: <https://www.km.kongsberg.com/ks/web/nokbg0239.nsf/obj/SBL-560x560.jpg/\protect\T1\textdollarFile/SBL-560x560.jpg?OpenElement>

O SBL apresenta algumas desvantagens, como a precisão ser proporcional ao número de *transceivers* utilizados, e o número de *transceivers* dependerá do tamanho da

embarcação utilizada, uma vez que o sistema é nela acoplado.

3.2.2.7 Ultra Short Baseline Positioning System

A *Ultra Short Baseline Positioning System* (USBL) é outro método para posicionamento utilizando ondas acústicas, apresentado na Figura 18. O método utiliza a comunicação entre o *transdutor* acoplado a embarcação e um ou mais *transponders* no robô. O *transdutor* envia uma onda acústica, e quando o *transponders* recebe e replica o mesmo sinal para o *transdutor*. Com esse sistema é possível calcular a posição relativa da embarcação para o robô.

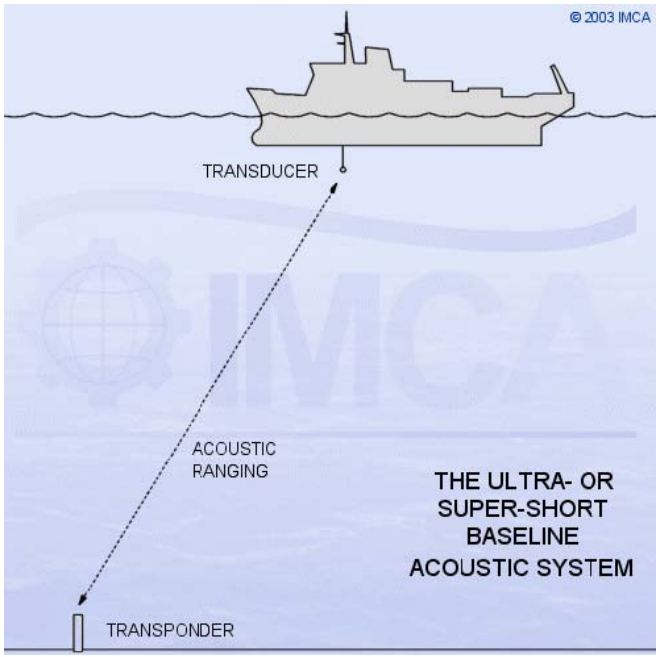


Figura 18 – Exemplo de como é realizada a comunicação em um sistema USBL

Fonte: <http://www.shipseducation.com/info/offshore/images/IMCA%20Introduction%20to%20Dynamic%20Positioning_img_8.jpg>

Watanabe, Ochi et al. (2012) demonstrou uma abordagem diferente utilizando o sistema do USBL, onde equipou o *transceiver* no AUV e o *transponder* na embarcação utilizada para descobrir a posição do AUV.

3.2.2.8 Long Base Line System

O *Long Base Line System* (LBL) é responsável por determinar a posição do robô utilizando a sua triangulação em relação à três ou mais *transponders*, como apresentado na Figura 19. Os *transponders* são precisamente fixados e geo-referenciados no fundo do mar, na forma de que o robô opere dentro do alcance deles.

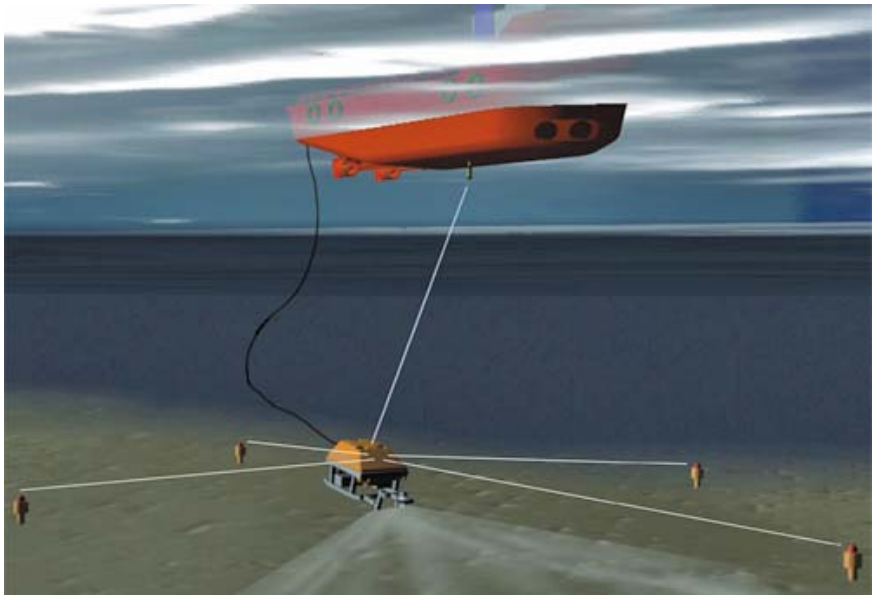


Figura 19 – Exemplo de como é realizada a comunicação em um sistema LBL

Fonte: <https://www.km.kongsberg.com/ks/web/nokbg0239.nsf/obj/cpap-500x340.jpg/\protect\T1\textdollarFile/cpap-500x340.jpg?OpenElement>

A principal vantagem desse sistema é a precisão, o que o torna atrativo para diversas aplicações em ambientes subaquático. Porém, a instalação dos *transponders* possui um custo elevado quando comparado com USBL e SBL.

3.2.2.9 Câmera

Nas abordagens SLAM terrestre e subaquática as câmeras são amplamente utilizadas para obtenção de informação do ambiente. As principais vantagens desse tipo de sensor são o custo empregado na aquisição quando comparado com SONARs de imageamento e LASERs, além da quantidade de informação capturada do meio. Porém, o sensor possui a limitação do processamento computacional das imagens e também da iluminação do ambiente, visto que, sua composição é baseada na incidência de luz nas partículas do ambiente visitado.

Devido à essa limitação da incidência da luz, o emprego das câmeras na robótica subaquática está atrelada a ambientes com boa visibilidade, ou empregando iluminação externa em tarefas em que a distância do robô ao alvo seja curta, como inspeção de dutos de petróleo, cascos de navio, perfil do fundo do mar, entre outros. Outro ponto importante, é que em abordagens autônomas, o robô deve ser capaz de processar os dados no menor tempo possível com o uso reduzido de energia.

Os tipos de câmera mais utilizados são as monoculares (Figura 20(a)) e estereoscópica (Figura 20(b)). As câmeras monoculares utilizam somente uma lente, enquanto as estereoscópica possuem mais de uma lente separadas por uma distância conhecida. A

distância entre as duas lentes ocasiona diferentes imagens, onde torna-se possível calcular a distância dos objetos para a câmera, similar a percepção que os humanos possuem. Também existem algumas abordagens que utilizam câmeras omnidirecionais.

(a) Câmera monocular utilizada em ambientes subaquáticos.

Fonte: <http://www.tritech.co.uk/media/products/6f8d2674b968b9725f0e52b7c2235fba_52912.jpeg>

(b) Câmera estereoscópica utilizada em ambientes subaquáticos.

Fonte: <http://www.bowtech.co.uk/iadmin/Uploads/productImages/1342447798.jpg>

Figura 20 – Exemplo das câmeras utilizadas para a percepção do ambiente em ambientes subaquáticos.

3.2.2.10 LASER

O LASER é um equipamento capaz de captar distâncias dos objetos existentes ao redor do robô. Apesar de ser amplamente utilizado em robótica terrestre, sua aplicação em ambientes subaquáticos é limitada devido ao uso de ondas de luz em sua operação (SILVEIRA, 2015).

No meio subaquático, tanto o LASER como a câmera sofrem com efeitos referentes a propagação da luz no meio. O *scattering* é a mudança de direção dos fótons na propagação da luz, tornando-se responsável por degradar as imagens nesse ambiente. Também ocorre a perda de energia de absorção durante a propagação no meio aquático. Nesse contexto, diferentes LASERs foram desenvolvidos para reduzir esses efeitos.

A principal desvantagem desse sensor, é a limitação do uso em distâncias pequenas e o alto custo. Um exemplo de LASER comercial é apresentado na Figura 21.

3.2.2.11 SONAR

Um SONAR é um dispositivo para detectar e localizar objetos remotamente na água utilizando o som (PAULL et al., 2014). O SONAR é classificado como ativo, quando o SONAR envia ondas sonoras especificas para o ambiente e espera o retorno delas, ou passivo, quando o SONAR fica adquirindo informações emitidas por objetos dentro do ambiente.

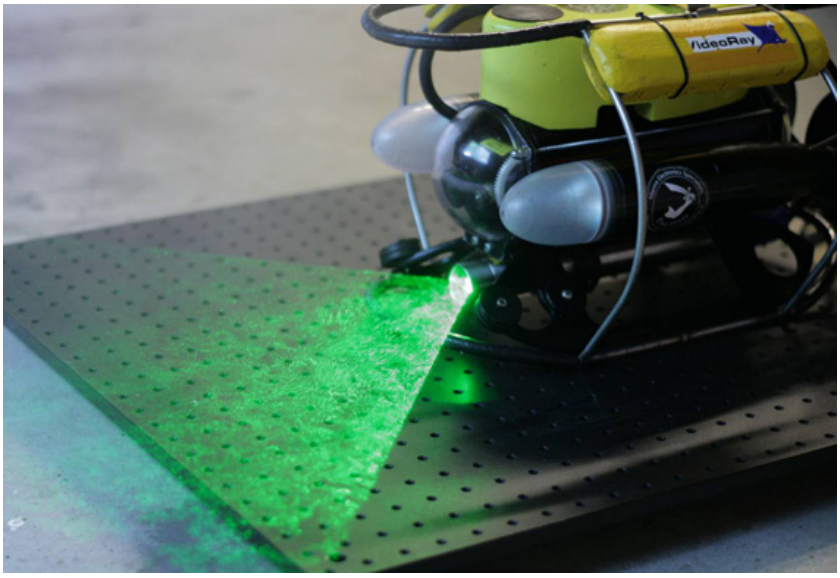


Figura 21 – Exemplo do uso do LASER em um ROV

Fonte: <http://www.savante.co.uk/wp-content/uploads/2010/01/savante-underwater-subsea-lumeneye-laser-profiler-jpg>

No SLAM subaquático o SONAR ativo é amplamente utilizado tanto para criar imagens representativas do fundo do mar quanto para criar mapas batimétricos. Nesse contexto, foram propostos diferentes métodos para a aquisição de dados, os quais serão explanados nas próximas seções.

3.2.2.11.1 Single Beam

O *Single Beam* utiliza a mesma ideia de funcionamento do altímetro (subseção 3.2.2.1), onde envia-se apenas um pulso para o fundo do mar e com o tempo de retorno é calculado a distância do veículo até o fundo do mar. Na Figura 22 é apresentado o seu princípio de funcionamento.

3.2.2.11.2 Multi Beam

O SONAR *Multi Beam* diferentemente do *Single Beam*, consegue mapear simultaneamente mais de um ponto de localização do fundo do oceano. O seu princípio de funcionamento baseia-se no envio de pulsos com ângulos diferentes a cada instante de medição. Na Figura 23 é mostrado o padrão de envio do *Multi Beam*.

3.2.2.11.3 Side Scan Sonar

O *Side Scan* é utilizado para criar uma imagem representativa do fundo do mar. Seu funcionamento utiliza como base o envio de ondas sonoras diretamente para baixo e

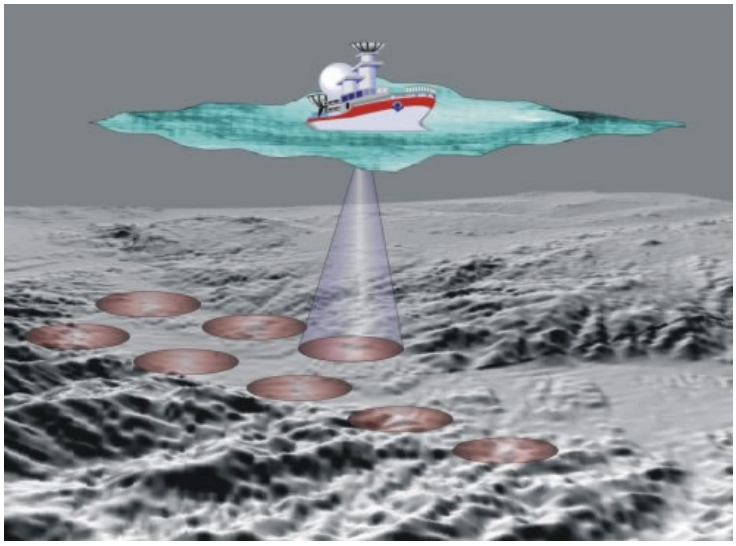


Figura 22 – Funcionamento do *Single Beam*

Fonte: <http://www.gebco.net/general_interest/faq/images/single_beam_data.jpg>

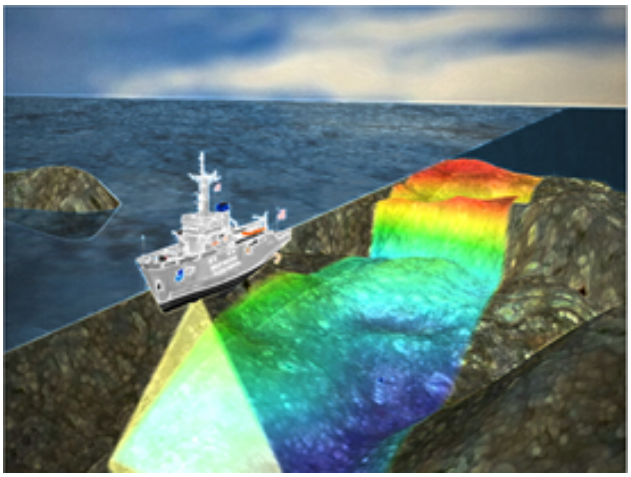


Figura 23 – Exemplo de funcionamento do *Multi Beam*

Fonte: <http://www.nauticalcharts.noaa.gov/staff/images/MultiBeam_movie_noaa.png>

para os lados em conjunto com a intensidade refletida deste sinal. A principal diferença entre o *Side Scan* e *Multi beam* é não fornecer dados sobre a profundidade da água. O seu princípio de funcionamento pode ser observado na Figura 24.

3.2.2.11.4 Mechanically Imaging Sonar

Mechanical Scanning Imaging Sonar (MSIS) (Figura 25) utiliza um *beam* acoplado a um atuador para fazer o *scan* do ambiente. Esse atuador é movimentado para cobrir os 360º do ambiente a partir de passos pré definidos. A cada passo é enviado um sinal, e através do retorno da amplitude e da distância é criada a imagem acústica. A principal

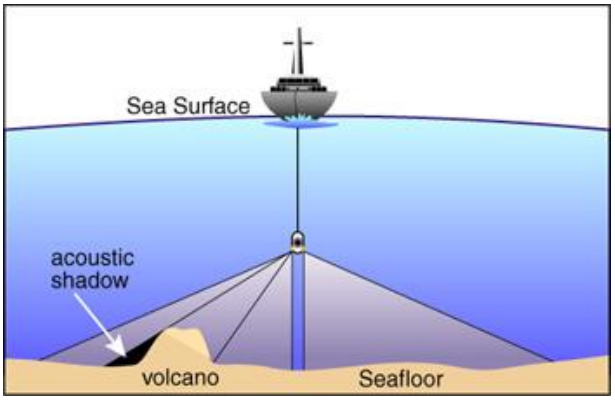


Figura 24 – Exemplo de funcionamento do *Side Scan*

Fonte: <http://www.punaridge.org/doc/factoids/digitaldata/ddfig2.jpg>

vantagem é o seu custo.

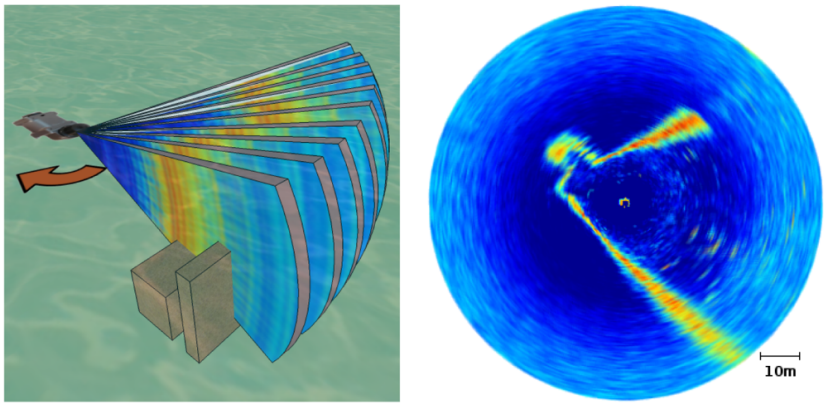


Figura 25 – Exemplo de funcionamento do MSIS retirado do artigo (BURGUERA; GONZÁLEZ; OLIVER, 2012). Na Figura (a) é apresentado o processo de escaneamento do MSIS, onde é demonstrado dois obstáculos para as ondas acústicas. A Figura (b) mostra a imagem acústica resultante, onde as cores apresentam as intensidades do sinal recebido.

3.2.2.11.5 Forward Looking Imaging Sonar

Forward-Looking Sonar (FLS) proporciona uma representação visual do ambiente em relação a frente do sensor. É um dispositivo que produz ondas acústicas que propagam-se pelo meio até colidirem com algum obstáculo ou ser totalmente absorvida (SANTOS et al., 2015). Com o sinal refletido é gerada a imagem do ambiente. O princípio de funcionamento do FLS é apresentado na Figura 26.

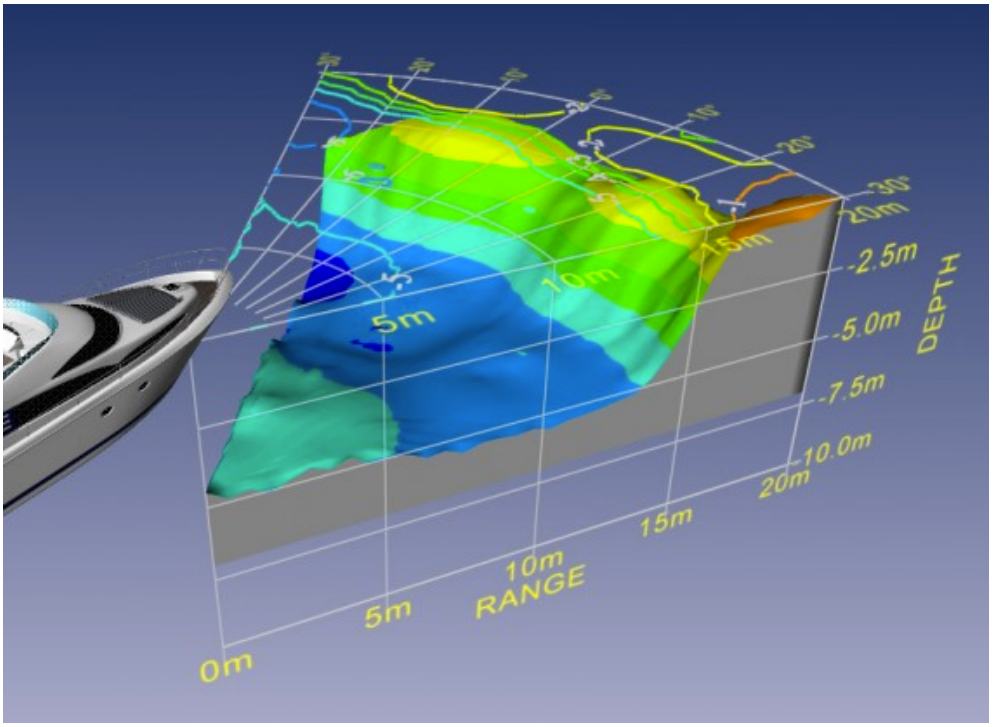


Figura 26 – Exemplo de funcionamento do fls

Fonte: <http://slingthehook.com/files/uploads/2012-10-07_12-42-21.png>

3.3 Revisão da Literatura em SLAM Subaquático

Williams et al. (2000) propôs uma abordagem utilizando o algoritmo EKF para estimar o deslocamento do Oberon AUV. Para os experimentos o AUV foi equipado com dois SONARs MSIS de baixa frequência, câmera, sensor de profundidade, giroscópio óptico. A extração das *features* é realizada a partir do SONAR, o algoritmo proposto foi testado com os dados coletados na piscina da Universidade de Sidney, e na costa da cidade de Sidney. Os resultados foram satisfatórios, porém o método possui a limitação de ser utilizado em ambientes com número reduzido de *features*, devido ao custo computacional ser proporcional ao seu número.

Em 2001, (LEONARD; CARPENTER; FEDER, 2001) apresentaram um algoritmo baseado em EKF para detectar e extrair *features* de objetos no fundo do mar utilizando um SONAR de imageamento frontal. Os dados utilizados foram coletados em uma expedição realizada em 1997, aonde utilizou-se uma plataforma acoplada ao navio e submersa em aproximadamente 5.5 metros. A plataforma foi equipada com SONAR, IMU, DVL para simular os subsistemas encontrados em AUV, e para o *ground truth* foi utilizado o DGPS. O algoritmo proposto apresentou resultados mais satisfatórios do que a estimativa utilizando os dados de *dead-reckoning*, porém o algoritmo possui complexidade de $\Theta(n^2)$ para as *features*.

Ainda em 2001, Ruiz et al. (2001) propôs um método utilizando *features* extraídas

com um MSIS. O algoritmo é baseado no EKF em conjunto com Multiple Hypothesis Tracking Filter (MHTF) para reduzir o número de hipóteses. Dois experimentos fora realizados: o primeiro utilizou um ambiente controlado com a informação de *ground truth*, e o segundo experimento foi realizado na faixa litorânea de Oban na Escócia. A proposta obteve bons resultados, porém o custo computacional aumenta exponencialmente em relação ao número de hipóteses.

Paul Newman e colaboradores publicaram dois trabalhos na área de SLAM subaquático. O primeiro trabalho (NEWMAN; LEONARD, 2003) apresenta uma proposta utilizando somente os dados provenientes dos *transponders* em um sistema LBL. Os dados foram coletados em águas rasas utilizando o Caribou AUV, equipado com DVL, bússola e um receptor do sistema LBL. O algoritmo desenvolvido foi comparado com um EKF e inicialmente obteve resultados consideráveis, entretanto, com o avanço do experimento a eficácia diminuiu devido as características do ambiente e do local da inicialização dos *transponders*.

O segundo trabalho (NEWMAN; LEONARD; RIKOSKI, 2005) propôs um algoritmo de SLAM em tempo constante utilizando um *Synthetic Aperture* SONAR para detectar pontos de referência no fundo do mar. O algoritmo utiliza sub-mapas locais para armazenar as *features* observadas dentro do limite do mapa gerado para mapa local respectivo. O algoritmo proposto mostrou resultados consistentes com a trajetória realizada.

Em 2004, Williams e Mahon (2004) apresentou um método para fusão dos dados oriundos da câmera ótica e SONAR em conjunto com EKF para o uso em áreas de corais. No método, as *features* visuais são rastreadas utilizando Lucas and Kanade feature tracking technique (LUCAS; KANADE et al., 1981). Após isso, foi realizada a reconstrução do ambiente utilizando a estimativa do SLAM.

Ruiz et al. (2004) propôs um algoritmo baseado em EKF utilizando o Sidescan SONAR. Os dados do Sidescan foram coletados através do REMUS AUV e a extração e associação das *features* são realizadas manualmente. O algoritmo demonstrou um erro menor após reconhecer um local anteriormente observado.

Ainda em 2004, foi proposto um algoritmo de Navegação Visual Aumentada (VAN) (EUSTICE; PIZARRO; SINGH, 2004). A trajetória do veículo é representada através de um *Augmented State Kalman filter* (ASKF), e o registro de imagens é realizado em pares devido a necessidade de se obter a similaridade entre dois quadros. O experimento realizado comparou o uso do ASKF com apenas os dados do *dead-reckoning* e com o algoritmo proposto em 100 imagens subaquáticas reais, onde o algoritmo proposto conseguiu encontrar 19 pontos semelhantes.

Roman e Singh (2005) propuseram um algoritmo utilizando EKF a partir dos dados provenientes de um SONAR Multibeam Batimétrico e do ROV JASON. O algoritmo cria

sub-mapas com os dados do SONAR e da posição estimada do robô, e posteriormente desenvolve um vínculo com os sub-mapas criados próximos segundo a medição relativa de movimento do robô. O resultado da abordagem foi comparado com os dados combinados do *dead-reckoning* e LBL, obtendo resultados consideráveis em pontos onde houve um maior número de sub-mapas sobrepondo-se.

Em 2006, foi proposto um método de SLAM utilizando minimização por entropia para criar mapas visuais 3D (SÁEZ et al., 2006). Para o experimento foi utilizado o robô AQUA equipado com câmera estéreo, e a odometria é realizada visualmente. A etapa de entropia é utilizada para melhorar a qualidade do algoritmo e é dividido em extração e *matching* de *feature*, refinamento e estimativa de transformação. O método apresentou resultados satisfatórios, entretanto o custo computacional é alto e o algoritmo atualmente é executado offline.

Ainda no mesmo ano, Olson, Leonard e Teller (2006) propuseram um método para eliminar possíveis erros de medida e um método de inicialização dos *transponders* num sistema LBL. Após a filtragem dos dados coletados é realizada uma etapa de votação para decidir entre duas posições possíveis do *transponder*. O método proposto foi adicionado em um EKF, encontrando resultados favoráveis quando comparado com o *ground truth*.

Uma extensão do FastSLAM (MONTEMERLO et al., 2002b) para ambientes baseado em Filtro de Partículas Rao-Blackwellized utilizando mapas 3D de ocupação foi proposto por Fairfield, Kantor e Wettergreen (2007). A proposta foi testada com dados coletados com o robô DEPTHX em uma área de caverna no México encontrando resultados relevantes. Também é realizado um comparativo entre o número de partículas para que a aplicação continue rodando em tempo real.

Uma proposta utilizando *Scale Invariant Feature Transform* (SIFT) e mapas topológicos foi proposto por (DREWS-JR; BOTELHO; GOMES, 2008) (BOTELHO; DREWS-JR; LEIVAS, 2008). O método utiliza SIFT para extrair os pontos mais significativos da imagem, após isso, é utilizado o *Random Sample Consensus* (RANSAC) e LMeds para remover os pontos considerados falsos-positivos. Também, os dados provenientes do SIFT são utilizados para compor e construir o mapa topológico utilizado *Self-Organizing Map* (SOM).

Em 2008, Eustice, Pizarro e Singh (2008) estendeu e apresentou com mais detalhes o trabalho publicado em (EUSTICE; PIZARRO; SINGH, 2004). O deslocamento do robô é calculado utilizando a odometria visual, e altímetro entre duas imagens que são consideradas consecutivas. É utilizado o *Extended Information Filter* para diminuir a complexidade computacional, e assim podendo utilizar em áreas maiores do que quando comparado com o EKF.

Outra abordagem de SLAM foi proposta em 2008, baseado em EKF para am-

bientes parcialmente estruturado utilizando MSIS (RIBAS et al., 2008). Nas imagens é utilizado para detectar as *features* o algoritmo de Hough Space. Também foi implementado mapas locais para diminuir o custo associado a atualização das matrizes utilizado no EKF. Os experimentos foram realizados em uma marina abandonada na Espanha utilizando o Ictineu AUV, e os resultados da abordagem foram superiores quando comparado com a imagem criada somente utilizando os dados proveniente do *dead-reckoning*.

Ainda em 2008, foi proposto um método de SLAM baseado em *Exactly Sparse Extended Information Filter* (ESEIF) utilizando um FLS para a inspeção de casco de navio (WALTER; HOVER; LEONARD, 2008). Para os testes de eficiência do algoritmo foram manualmente selecionadas as *features* e realizado um comparativo entre a proposta, EKF e *ground truth*, onde a proposta conseguiu preservar a consistência relativa ao EKF. O autor afirma que o método possui resultados satisfatórios utilizando as *features* selecionadas manualmente, porém utilizando um *framework* para detecção automática o método perde a sua eficácia.

Mallios et al. (2010) desenvolveu um método utilizando dois algoritmos EKF em conjunto com *scan matching* probabilístico nas imagens capturadas através de um MSIS. O objetivo do *scan matching* é calcular o deslocamento relativo do veículo em duas imagens consecutivas. Como citado anteriormente, o método utiliza dois algoritmos de EKF executados em paralelo, onde o primeiro é responsável por estimar o *dead-reckoning* e o segundo utiliza os dados referente a probabilidade do *scan matching*. A performance do algoritmo foi testada utilizando o *dataset* disponibilizado por (RIBAS et al., 2008), e os resultados quando comparados com o *dead-reckoning* apresentaram uma significativa melhora na correção do mapa.

Uma abordagem para criar o mapa utilizando MSIS e EKF com as coordenadas fixadas no robô foi proposto em 2011 (BURGUERA; GONZÁLEZ; OLIVER, 2011). Segundo o autor, a vantagem do uso das coordenadas fixadas no robô é a redução do número de linearizações no EKF. Assim, a proposta do autor foi comparada com a abordagem das coordenadas fixadas no mundo utilizando os dados referente a (RIBAS et al., 2008), onde a abordagem do autor foi superior.

Em 2012, foi apresentado um método para explorar ambientes desconhecidos e criar mosaicos referentes ao fundo do mar (FERREIRA et al., 2012). O método utiliza EKF em conjunto com as *features* extraídas utilizando o algoritmo *Speeded Up Robust Features* (SURF). Também foi utilizada a abordagem de criar mosaicos locais, e assim, possibilitando o uso em tempo real e melhores resultados no mosaico final (conjunto de mosaicos locais).

Barkby et al. (2012) propôs um SLAM batimétrico utilizando Rao-Blackwellized Particle Filter em conjunto com regressão de processo gaussiano para reconhecimento de lugares. O autor apresenta um comparativo da abordagem utilizando o *dead-reckoning* em

conjunto com USBL, mapa de grade (*grid map*) e mapa de trajetória (*trajectory map*). As duas abordagens podem ser igualmente consideradas eficientes, enquanto a baseada em mapa de grade apresenta uma maior complexidade de tempo de execução e a baseada na trajetória apresenta uma menor complexidade de espaço.

Um método de SLAM visual em tempo real para a inspeção de casco foi proposto em (KIM; EUSTICE, 2013). O autor propõem o uso do algoritmo *Pose-Graph* Visual SLAM em conjunto com o algoritmo incremental smoothing and mapping (iSAM) (KAESS; RANGANATHAN; DELLAERT, 2008). Uma etapa é realizada para melhorar as imagens armazenadas utilizando uma equalização por contraste, após isso, é utilizado SURF e *Bag Of Words* (BOW) para a descrição da imagem. Por último é mensurado a distância entre duas imagens e adicionado esse valor no iSAM.

Mallios et al. (2014) apresentou um algoritmo para resolver o problema de SLAM utilizando *Scan Matching* para ambientes subaquático. O algoritmo utiliza dois EKFs como apresentado em (MALLIOS et al., 2010) e foi comparado com SLAM baseado em *features* e testado utilizando dados de um *dataset* real. O resultado da abordagem foi similar ao SLAM visual.

Ainda em 2014, uma proposta baseada em um *Single Cluster Probability Hypothsis Density* (SC-PHD) foi apresentada para resolver o problema do SLAM (LEE et al., 2014). O processo de agrupamento é baseado na relação hierárquica entre dois pontos dentro do processo. A hierarquia é utilizada com os dados do mapa de *features* e dos dados do estado atual do veículo. Os testes foram realizados em um tanque de experimentos, onde os algoritmos SC-PHD e Rao–Blackwellized-PHD foram comparados, e a proposta obteve melhores resultados.

Um Side Scan Sonars SLAM (SSS-SLAM) foi apresentado por Moreno, Burguera e Oliver (2014). O método propõem um *framework* para retornar a trajetória georeferenciada do AUV e criar mosaicos acústicos. Para isso, utiliza um algoritmo EKF com as características de padronização de dados, facilidade no uso de dados de sensores como IMU ou GPS. A proposta foi apresentada em um ambiente real, utilizando um AUV equipado com DVL e a posição de GPS quando o robô está na superfície.

Bonin-Font et al. (2015) apresentaram um comparativo entre o EKF e o Graph-SLAM baseado em visão estéreo. O primeiro experimento realizado utilizou o robô Girona500 equipado com uma câmera dentro de um tanque de água. No tanque foi colocado uma foto no fundo para simular um fundo real. O deslocamento do robô é calculado utilizando odometria visual através da *LibViso2*. O segundo experimento utiliza o micro-AUV em uma área costeira. Foram utilizadas marcadores artificiais para prover uma pose inicial/final. Em ambos os experimentos, a profundidade é constante e o algoritmo Graph-SLAM obteve performance levemente melhor que o EKF.

Um algoritmo de SLAM utilizando visão monocular e recuperação através de imagem-para-imagem foi proposto (HONG; KIM; KIM, 2015). O algoritmo para *loop closure* possui três passos, o primeiro é utilizar a distância de *Mahalanobis* para diminuir a possibilidade de imagens similares, o segundo passo utiliza SURF associado com BOW e um ajuste final para otimizar as poses relativas. No último passo é realizado uma técnica de ajuste para observar as melhores poses. Para o experimento foi utilizado um tanque de água, um *hover-capable* AUV equipado com IMU, *Attitude and Heading Reference System* (AHRS), DVL, scanning SONAR, altímetro, USBL e câmera. O erro encontrado foi menor que a abordagem sem a recuperação através de imagem-para-imagem.

Durante o ano de 2015, os mesmos autores propuseram um método robusto para *loop-closure* em SLAM visual subaquático (HONG et al., 2015). O método é mais detalhado e comparado com o algoritmo ASKF. O primeiro teste foi realizado em um tanque de água, obtendo a redução do erro quando comparado com o ASKF. O segundo experimento foi proposto em um ambiente real, e novamente encontrou melhores resultados na comparação, porém no segundo experimento, o tamanho do erro apresentou valor maior quando comparado com o primeiro.

Um sistema de localização baseado em visão para AUV foi proposto em Burguera, Bonin-Font e Oliver (2015). Essa proposta possui as características de reduzir o custo computacional e os erros de linearização em um EKF tradicional. Os autores propõe um esquema baseado em trajetória utilizando SIFT e RANSAC para eliminar os pontos associados erroneamente. O experimento foi realizado utilizando ambientes simulados, tanque de água e ambiente real obtendo ótimos resultados para estimativa de pose.

Silveira et al. (2015) apresentou o algoritmo DolphinSLAM, uma abordagem bioinspirada para SLAM 3D. O método é uma extensão do RatSLAM (MILFORD; WYETH, 2008) possuindo integração com o *Robot Operating System* (ROS) (QUIGLEY et al., 2009) para ambientes subaquáticos. Possui suporte para imagens ópticas – utilizando SIFT ou SURF e imagens acústicas. Os resultados foram promissores utilizando o *Underwater Simulator* (UWSIM) (PRATS et al., 2012), entretanto em ambientes reais o método não encontrou resultados satisfatórios. Na seção 4.1, o método é explanado detalhadamente devido ao seu uso na proposta deste trabalho.

Uma abordagem probabilística foi proposta utilizando o SONAR multibeam para SLAM 3D (PALOMER; RIDAO; RIBAS, 2016). A abordagem utiliza EKF e duas etapas para registrar as imagens –ponto-a-ponto e ponto-a-plano – utilizando uma estrutura *octree* em conjunto com nuvem de pontos e sub-mapas. Os experimentos foram baseados em dois *datasets* reais, onde os resultados indicaram melhor performance quando comparado com os dados provenientes do *dead reckoning* e USBL. Também, uma heurística foi implementada para diminuir a complexidade de $O(n^2)$ para $O(n)$ na etapa de associação.

Massot-Campos et al. (2016) propõem um SLAM *pose-graph* batimétrico utili-

zando nuvem de pontos coletadas através de um sistema de luz estruturada. O experimento foi avaliado utilizando dois dataset: simulado e um real. O artigo apresenta um erro menor que o *dead reckoning*, focando na criação dos submapas em relação ao ambiente, porém não apresenta o mapa para realizar o comparativo da trajetória.

Um SLAM utilizando fontes acústicas é proposto por (CHOI et al., 2016). Na proposta é utilizado um algoritmo EKF em um robô equipado com dois hidrofones, onde o robô precisa localizar ele mesmo e os emissores de sinal acústico. Para o experimento foi utilizado dados coletados em uma marina, utilizando um veículo equipado com DVL, IMU, hidrofone e DGPS. Os resultados encontrados pelo EKF foram condizentes com a trajetória encontrada pelo DGPS.

Chaves et al. (2016) apresentam uma proposta de SLAM visual para a inspeção de cascos de navios. Na proposta é apresentado um algoritmo para trajetória em conjunto com o SLAM para realizar a visitação de um lugar previamente visitado. Para isso, utilizam algoritmo de *Pose-graph* em dados coletados através do Hovering AUV. O algoritmo encontrou resultados superiores em relação a outras abordagens, com a vantagem da predição, habilidade na procura dos caminhos possíveis.

A Tabela 1 apresenta uma síntese dos trabalhos encontrados na literatura onde apresenta-se a informação relativa ao ano do trabalho, autor, robô, sensores, algoritmo e ambiente de teste. As propostas utilizam majoritariamente EKF ou filtro de partículas, sendo apresentadas algumas alternativas para reduzir o custo computacional destes algoritmos, através da limitação do número de *features*, uso de sub-mapas. Para a validação das propostas são utilizados dados reais e simulados onde são realizados comparativos em relação ao *ground truth* e *dead reckoning*. Nesse contexto de abordagens majoritariamente probabilísticas, as abordagens bioinspiradas entram como uma alternativa para este tipo de ambiente.

Neste capítulo apresentou-se os desafios encontrados e os sensores empregados no meio subaquáticos. Os trabalhos propostos para a resolução do problema de SLAM foram apresentados e na Tabela 1 as informações referentes à tipo de robô, algoritmo, ambiente e sensores. No Capítulo 4 serão apresentados os métodos utilizados neste trabalho.

Tabela 1 – Revisão do robô, tipo de ambiente, sensores e algoritmos utilizados.

Ano	Autor	Robô	Sensores	Algoritmo	Ambiente de teste
2000	(WILLIAMS et al., 2000)	Oberon AUV	Sensor de Pressão, giroscópio, MSIS	EKF	Real
2001	(LEONARD; CARPENTER; FEDER, 2001)	Plataforma de teste	DVL, IMU, DGPS	EKF	Real
2001	(RUIZ et al., 2001)	Não mencionado	SONAR	EKF e MHTF	Real e tanque de água
2004	(RUIZ et al., 2004)	Remus AUV	LBL, Sidescan SONAR	EKF	Real e Simulado
2004	(NEWMAN; LEONARD, 2003)	Caribou AUV	LBL, bússola, DVL, DGPS	EKF	Real
2004	(WILLIAMS; MAHON, 2004)	Oberon	Sensor de profundidade, giroscópio, bússola, duas câmeras,	EKF	Real
2004	(EUSTICE; PIZARRO; SINGH, 2004)	scientific AUV	Tilt, Bússola, AHRS, Altímetro, Sensor de pressão, DVL	EKF	Real
2005	(ROMAN; SINGH, 2005)	ROV Jason	Multibeam SONAR, DVL, Altímetro, Sensor de Pressão, LBL	EKF	Real
2005	(NEWMAN; LEONARD; RIKOSKI, 2005)	AUV	Synthetic Aperture sonar, LBL, DVL bússola, Inclinômetros, sensor de profundidade	EKF	Real
2006	(SÁEZ et al., 2006)	AQUA robot	câmera stereo, IMU.	6DOF egomotion algorithm Minimization	Real
2006	(OLSON; LEONARD; TELLER, 2006)	Odyssey III AUV	LBL	EKF	Real
2007	(FAIRFIELD; KANTOR; WETTERGREEN, 2007)	DEPTHX Vehicle	IMU, DVL, pencil-beam SONAR	Baseado no FastSLAM	Tanque de teste e Real
2008	(EUSTICE; PIZARRO; SINGH, 2008)	SeaBED AUV	Tilt, Bússola, AHRS, Altímetro, Sensor de Pressão, DVL	EKF	Real e Tanque de Teste
2008	(RIBAS et al., 2008)	ICTNEU Robot	DVL, giroscópio, MSIS, DGPS	EKF	Real
2008	(DREWS-JR; BOTELHO; GOMES, 2008)	ROVFURGII	Camera, miniking sonar, Altímetro e acelerômetro)	Self-organization mapping	Tanque de água
2008	(WALTER; HOVER; LEONARD, 2008)	Hovering AUV	DVL, IMU, Sensor de profundidade, bússola, SONAR DIDSON	ESEIF	Real – Inspeção de casco.
2010	(MALLIOS et al., 2010)	ICTNEU Robot	DVL, giroscópio, MSIS, DGPS	EKF and ASEKF	Real
2011	(BURGUERA; GONZÁLEZ; OLIVER, 2011)	Ictineu AUV	MSIS, DVL, bússola, giroscópio, Unidade de referência de movimento	EKF	Real
2012	(FERREIRA et al., 2012)	Romeo ROV	LASER triangulation altimeter	EKF	Real
2012	(BARKBY et al., 2012)	Sirius AUV, ROV Jason, Sentry AUV	Inclinometers,DVL, USBL, Multibeam SONAR, LBL	RBPF e modelo gaussiano	Real
2013	(KIM; EUSTICE, 2013)	Bluefin Robotics HAUV	DVL, IMU, Sensor de Pressão, Profundidade, Câmera	Pose-Graph Visual	Real
2014	(MALLIOS et al., 2014)	AUV	MSIS	EKF	Real
2014	(LEE et al., 2014)	Girona500	DVL, IMU, Câmera	SC-PHD	Real, tanque no CIRS
2014	(MORENO; BURGUERA; OLIVER, 2014)	AUV	IMU, DVL, GPS	EKF	Real
2015	(BONIN-FONT et al., 2015)	AUV	Camera stereo	EKF, Graph-SLAM	Real
2015	(HONG; KIM; KIM, 2015) (HONG et al., 2015)	Hover-capable AUV	IMU, AHRS, DVL, scanning SONAR, altímetro, USBL, câmera	ASKF	Real, Tanque de água
2015	(BURGUERA; BONIN-FONT; OLIVER, 2015)	Fugu-C mini-AUV	camera, profundidade,	EKF and IEKF	UWSIM, Real
2015	(SILVEIRA et al., 2015)	G500 simulado	DVL, IMU, Câmera	bioinspirado	Simulado no UWSIM
2016	(PALOMER; RIDAO; RIBAS, 2016)	Sirius AUV, Girona 500 AUV	DVL, AHRS, sensor de pressão, multibeam SONAR, SONAR	EKF	Real
2016	(MASSOT-CAMPOS et al., 2016)	ROV Hyper-Dolpin	LASER e câmera monocromática	Pose-graph	Simulado e real
2016	(CHOI et al., 2016)	Não mencionado	DVL, IMU, DGPS hidrofone	EKF	Real
2016	(CHAVES et al., 2016)	Hovering AUV	DVL,sensor de profundidade, SONAR e camera	Pose-graph	Real e simulação hibrida

4 Métodos

Neste capítulo serão detalhados os módulos utilizado pelo algoritmo DolphinSLAM, proposto pelo grupo e utilizado para SLAM subaquático. Também apresenta-se o modelo de CANN implementado neste trabalho.

4.1 DolphinSLAM

DolphinSLAM é um algoritmo para a resolução do problema de SLAM subaquático desenvolvido por (SILVEIRA et al., 2015) o qual utiliza como inspiração o comportamento dos neurônios dos mamíferos durante a etapa de localização e mapeamento.

O algoritmo DolphinSLAM é baseado no método RatSLAM (MILFORD; WYETH, 2008) o qual obteve resultados satisfatórios no meio terrestre e foi explanado na subseção 2.3.2.4. As principais mudanças realizadas no DolphinSLAM em comparação com o RatSLAM referem-se a percepção e a rede neural.

O módulo de percepção foi redefinido para suportar imagens acústicas e imagens ópticas, e também para integrar o algoritmo FAB-MAP (CUMMINS; NEWMAN, 2008) para o reconhecimento de locais semelhantes. Entretanto neste trabalho não será utilizado o algoritmo FAB-MAP devido ao aumento do custo computacional, e também pelos testes iniciais não apresentarem ganho considerável quando comparado ao algoritmo BOW para o reconhecimento.

A rede neural passou a utilizar a função do tipo Chapéu Mexicano, para aumentar a eficiência do algoritmo, uma vez que não é necessário uma etapa de inibição lateral após a excitação da rede. Também devido à característica do ambiente tridimensional foi proposto o uso dos neurônios com comportamento similar ao encontrado nos morcegos e demonstrado por Yartsev e Ulanovsky (2013).

A integração de caminho foi estendida para compreender o movimento nas três dimensões, e uma etapa de normalização da rede é utilizada para manter os neurônios com a ativação entre zero e um, seguindo o trabalho publicado por Stringer et al. (2002a).

O algoritmo foi desenvolvido utilizando a linguagem $C++$, e o ROS (QUIGLEY et al., 2009) como sistema principal devido ao seu grande uso na área da robótica. O código foi escrito utilizando bibliotecas de código aberto, como a biblioteca de visão computacional OpenCV (BRADSKI; KAEHLER, 2008) e algoritmos da biblioteca BOOST (SCHÄLING, 2011).

Uma outra importante característica do algoritmo DolphinSLAM é a sua estru-

tura modular como demonstrado na Figura 27, facilitando modificações e correções no algoritmos. Nas próximas subseções serão apresentados cada módulo do algoritmo.

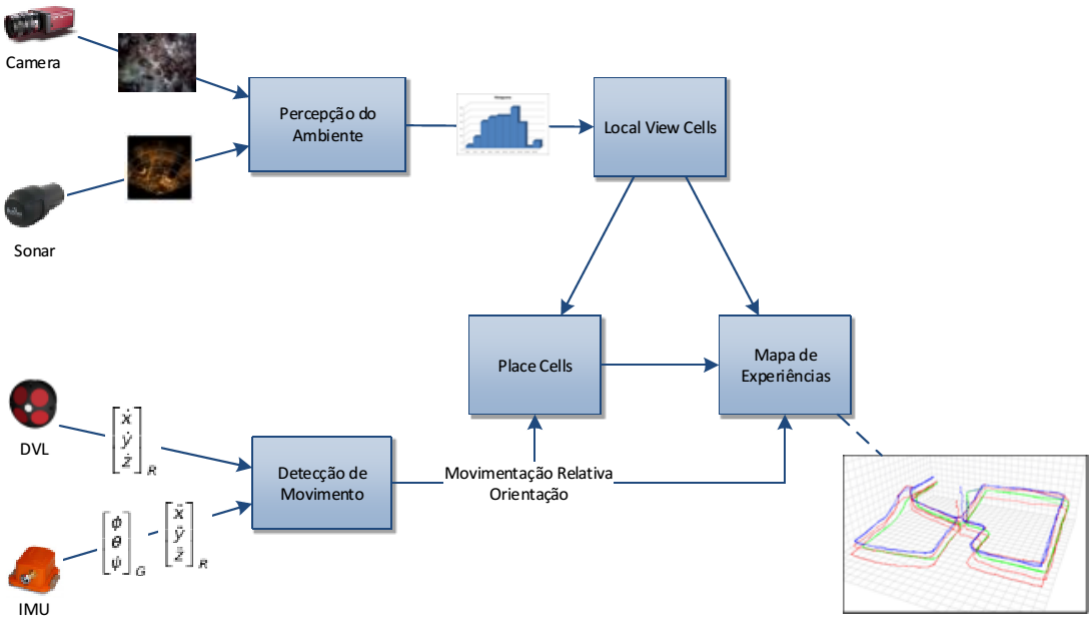


Figura 27 – Arquitetura do Algoritmo DolphinSLAM. Imagem extraída de (SILVEIRA, 2015).

4.1.1 Percepção do Ambiente

O módulo de percepção do ambiente é uma etapa importante em uma abordagem para a resolução do problema de SLAM, uma vez que a partir da correspondência destes dados decorre a correção da trajetória do robô. Nesta etapa, os dados provenientes dos sensores acústicos e ópticos são processados e enviados para o módulo da *Local View Cells*.

Atualmente, o DolphinSLAM suporta os dados de câmeras monoculares e de SONAR do tipo FLS (SANTOS, 2016). O uso das câmeras são indicados para lugares com baixo nível de turbidez e um alto nível de *features* significativas. Em contrapartida, os SONARs fornecem dados significativos em ambientes com alta turbidez.

4.1.1.1 Imagens Ópticas

Como mencionado na subseção 3.2.2.9, a captura de uma imagem do tipo óptica é dependente da iluminação do ambiente e da turbidez da água. Porém, o seu uso continua em evidência devido às diversas aplicações no meio subaquático, como os estudos dos organismos subaquáticos, inspeção de cascos de navios e de dutos de petróleo, entre outros.

Nesse contexto, são utilizadas técnicas responsáveis por extrair as principais *features* de uma imagem. No algoritmo DolphinSLAM está implementado o uso dos descritores

SIFT (LOWE, 2004) e SURF (BAY; TUYTELAARS; GOOL, 2006). Em (ZAFFARI et al., 2016), foi realizado um estudo sobre os descritores em datasets simulados através do UWSIM (PRATS et al., 2012), onde encontrou-se melhores resultados tanto em desempenho quanto em correspondência de frames utilizando o descritor SURF.

4.1.1.2 Imagens Acústicas

As imagens acústicas possuem alguns desafios no seu uso, como a baixa resolução, ruídos, distribuição sonora não uniforme, sombra acústica, entre outros (SANTOS, 2016). Entretanto possui a característica de prover formas geométricas, as quais podem ser extraídas utilizando detectores específicos.

A extração dos dados através das formas geométricas utiliza o cálculo dos sete momentos de Hu (HU, 1962). Para aprimorar a qualidade dos dados, primeiramente realiza-se uma filtragem utilizando o filtro Gaussiano, após essa etapa utiliza-se um filtro mediano, e por último as imagens são binarizadas para melhorar a detecção dos contornos.

Atualmente, encontra-se em fase de estudos a utilização de um método baseado na similaridade de grafos topológicos extraídos das imagens acústicas (SANTOS et al., 2015; SANTOS, 2016) para detecção nas imagens acústicas.

4.1.1.3 Bag-Of-Words

Ambos os sensores utilizados para a percepção do ambiente adotam a representação das imagens a partir do algoritmo BOW proposto em (CSURKA et al., 2004). O algoritmo é responsável por criar um histograma representativo das *features* extraídas da imagem. Na Figura 28 são apresentadas as etapas do algoritmo, e posteriormente cada módulo é explicado:

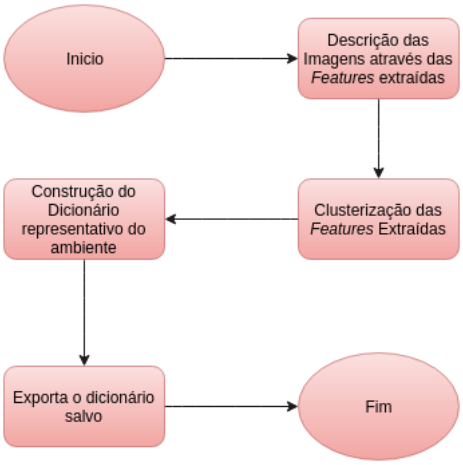


Figura 28 – Etapas utilizadas pelo BOW.

Primeiramente são utilizadas imagens pré-definidas para o treinamento do algoritmo, onde são extraídas *features* através dos descritores (SIFT ou SURF). Posterior-

mente, é realizada a clusterização desses descritores utilizando a técnica de *K-means* (ARTHUR; VASSILVITSKII, 2007), onde as *features* são agrupadas conforme a sua similaridade. Estes grupos de similaridades serão responsáveis por gerar as "palavras" utilizadas pelo dicionário utilizado. Após concluído o dicionário, cada imagem é descrita através de um histograma calculado em cima do dicionário.

O tamanho do dicionário é outro fator que pode influenciar no reconhecimento de locais semelhantes. Dicionários grandes possuem a tendência de classificar as imagens sendo diferentes, enquanto dicionários pequenos classificam sendo muito iguais. Nesse contexto, em (ZAFFARI et al., 2016), foi realizado um estudo entre o tamanho dos dicionários utilizados em imagens correspondentes, e assim, demonstrando os efeitos encontrados no reconhecimento de locais.

4.1.2 Detecção de Movimento

No módulo Detecção de Movimento são integrados os dados referentes aos sensores proprioceptivos e exteroceptivos de movimento do robô. Devido às características do meio subaquático são utilizados sensores específicos como demonstrado na Seção 3.2. Para calcular o movimento do robô é necessária a definição de um sistema de coordenadas global, conforme Silveira (2015). Ele é definido como:

- O eixo x acompanha o sentido do norte magnético da terra, quando esta informação está disponível. Caso contrário, é fixado na primeira posição do robô, em concordância com seu sistema de coordenadas.
- O eixo y é definido com seu eixo crescente na direção leste. Da mesma forma, na ausência de leitura magnética da terra, o mesmo é fixado em cima do eixo y do robô, em sua primeira leitura.
- O eixo z em concordância com a regra da mão direita, possui sua direção crescente apontado para baixo, em direção ao fundo do mar.
- ϕ chamado de *roll*, refere-se a rotação em torno do eixo x.
- θ chamado de *pitch*, refere-se a rotação em torno do eixo y.
- ψ chamado de *yaw*, refere-se a rotação em torno do eixo z.

O principal sensor utilizado para a medição de velocidade é a DVL, o seu princípio de funcionamento baseia-se no envio de ondas sonoras em direção ao fundo do mar. Entretanto, a velocidade calculada é em relação ao robô, fazendo-se necessário o uso de outro sensor para a orientação do robô, como os magnetômetros ou IMU.

No algoritmo DolphinSLAM está configurado o tratamento de dados para a DVL e IMU. Primeiramente é realizada a transformação dos dados para a orientação em ângulos de euler (ϕ, θ e ψ) conforme demonstrado na Equação 4.1

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}^G = R(\phi, \theta, \psi) \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}^R, \tag{4.1}$$

onde a matriz R refere-se a matriz de rotação responsável por transformar os dados do robô para os dados globais, e a matriz G aos dados no sistema global.

A captura de imagens é utilizada como gatilho para a atualização do sistema DolphinSLAM, podendo ser atualizada de acordo com alguns *keyframes* escolhidos. Com isso, o deslocamento deve ser acumulado entre o tempo de dois *keyframes* capturados (SILVEIRA, 2015).

Assim, faz-se necessário o uso de uma métrica para acumular os dados enquanto uma nova imagem não é recebida. Primeiramente é calculado a partir da Equação 4.2 o deslocamento instantâneo:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}^G = \begin{bmatrix} x \\ y \\ z \end{bmatrix}^G_t + \Delta t \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}^G_t, \tag{4.2}$$

onde Δt é o tempo entre duas leituras da DVL. Quando detectado um novo *keyframe*, este valor acumulado é repassado para os módulos *Place Cells* e Mapa de Experiência e posteriormente reinicializado. A partir da soma dos dados de movimentação é possível criar o mapa de *Dead Reckoning*, utilizado para a validação do algoritmo.

4.1.3 Local View Cells

As *Local View Cells* são os neurônios responsáveis pela percepção do ambiente. As imagens processadas no módulo de Percepção do Ambiente são encaminhados para o módulo da *Local View*. Toda vez que ocorre a aquisição de uma nova imagem pelo *Local View*, ela é comparada através do histograma gerado pelo BOW com os histogramas das outras imagens previamente armazenadas, conforme apresentado na Equação 4.3:

$$R_{similar} = compare(A, B_i), \tag{4.3}$$

onde, $R_{similar}$ refere-se a taxa de comparação dos histogramas das duas imagens, A é a imagem atual, e B_i a imagem armazenada na *Local View* comparada (l_i). Caso a

similaridade da imagem ($R_{similar}$) ultrapasse um limiar previamente definido (L_{limiar}), a *local view* com a maior taxa de similaridade é ativada:

$$\begin{cases} R_{similar} > L_{limiar} = l_i, \\ R_{similar} < L_{limiar} = l_m, \end{cases} \tag{4.4}$$

onde l_i é a *local view* comparada e consequentemente ativada quando $R_{similar}$ maior que o limiar, e l_m a *local view* criada e adicionada ao conjunto de *local views* (Equação 4.5) quando o $R_{similar}$ possuir valor menor que o limiar:

$$L = \{l_1, l_2, l_3, \ldots, l_i, \ldots, l_m\}. \tag{4.5}$$

A cada instante de tempo, somente uma das células armazenadas pelo algoritmo estará ativa.

4.1.4 Place Cells

A *Place Cell* (subseção 2.3.2.1) proposta por Silveira (2015), utiliza como base a descoberta da existência de *Place Cells* tridimensionais no cérebro dos morcegos por Yartsev e Ulanovsky (2013). O algoritmo proposto utiliza as três dimensões para representar o ambiente, nesse contexto, escolheu-se o modelo computacional de neurônios do tipo *Place Cell 3D*.

A modelagem dos neurônios é realizado através do uso de uma CANN. Segundo Stringer et al. (2002a), a CANN mantêm as taxas de ativação dos neurônios para representar um ambiente, com a característica de utilizar ligações recorrentes excitatórias entre os seus elementos para refletir a distância entre os neurônios no ambiente. Outro ponto importante é a definição do pacote de atividade neural sendo o conjunto de neurônios ativos que representam uma localização específica do meio ambiente.

A estrutura da CANN é circular, ou seja, entre os neurônios de uma extremidade a outra, assim não limitando a capacidade de mapeamento pelo número de neurônios. Outra característica é a presença de ligação sináptica entre todos os elementos, tornando a ativação de um neurônio igual à soma de todos os valores das ligações sinápticas de todos os outros neurônios a ele conectados.

O processo de atualização da CANN é dividido em quatro etapas: excitação Lateral, integração de caminho, excitação externa e normalização. Nas próximas subseções serão explanados cada etapa implementada pelo algoritmo DolphinSLAM.

4.1.4.1 Excitação Lateral

Neurônios próximos na rede cooperam para a criação de um pacote de ativação, enquanto neurônios afastados competem pela manutenção de seu pacote de ativação (SILVEIRA, 2015). Na CANN todos os neurônios possuem ligações sinápticas entre si, onde o peso das ligações é calculado através da distância dos neurônios. Essas ligações podem ocasionar valores positivos (atração) e valores negativos (inibição).

A atividade da rede neural é calculada utilizando a Equação 4.6 e Equação 4.7, onde as taxas de ativação e inibição são proporcionais a distância entre cada par de neurônios da rede, e assim, utilizando as taxas de ativação realimenta-se a rede neural. Para calcular a taxa de excitação utiliza-se as seguintes Equações:

$$r_{x',y',z'} = \sum_{a=1}^{n_{x'}-1} \sum_{b=1}^{n_{y'}-1} \sum_{c=1}^{n_{z'}-1} \epsilon(d_{x'}, d_{y'}, d_{z'})\, r_{a,b,c} \quad \forall x', y', z', \tag{4.6}$$

onde

$$\begin{aligned} d_{x'} &= \min(|x' - a|, n_{x'} - |x' - a|), \\ d_{y'} &= \min(|y' - b|, n_{y'} - |y' - b|), \\ d_{z'} &= \min(|z' - c|, n_{z'} - |z' - c|), \end{aligned} \tag{4.7}$$

e a tripla (x', y', z') representa as coordenadas dos neurônios de interesse, a tripla (a, b, c) representa a coordenada de qualquer outro neurônio da CANN, $n_{x'}$, $n_{y'}$, e $n_{z'}$ a quantidade de neurônios de cada dimensão da CANN, $d_{x'}, d_{y'} e d_{z'}$ referem-se a distância entre os neurônios, $\epsilon(d_{x'}, d_{y'}, d_{z'})$ o peso sináptico, e $r_{a,b,c}$ é a taxa de ativação (ou inibição) do neurônio com as coordenadas da tripla (a, b, c). O valor do peso sináptico ϵ é proporcional a distância entre o par de neurônios correspondentes.

No algoritmo DolphinSLAM, é possível escolher entre a Função Chapéu Mexicano e a Função Gaussiana para calcular os pesos sinápticos. A função Gaussiana é descrita pela Equação 4.8:

$$\epsilon(d_{x'}, d_{y'}, d_{z'}) = e^{-\left(\frac{d_{x'}^2 + d_{y'}^2 + d_{z'}^2}{\sigma^2}\right)}, \tag{4.8}$$

A função Chapéu Mexicano possui a característica de excitação forte nas ligações sinápticas entre neurônios próximos, ligações sinápticas inibitórias para os neurônios localizados a média distância, e em neurônios a longa distância um comportamento sináptico nulo, ou seja, nem excitatório nem inibitório. Os pesos sinápticos são apresentados na Equação 4.9:

$$\epsilon(d_{x'}, d_{y'}, d_{z'}) = \left(1 - \frac{d_{x'}^2 + d_{y'}^2 + d_{z'}^2}{\sigma^2}\right) e^{-\left(\frac{d_{x'}^2 + d_{y'}^2 + d_{z'}^2}{2\sigma^2}\right)}, \tag{4.9}$$

Em ambas as equações, σ refere-se ao desvio padrão da função do Chapéu Mexicano e da Função Gaussiana.

4.1.4.2 Integração de Caminho

O DolphinSLAM realiza a integração de caminho similarmente ao proposto pelo algoritmo do RatSLAM, com a diferença do tratamento dos dados para a dimensão z. A integração de caminho ocorre através do deslocamento da cópia do pacote de ativação utilizando os dados provenientes da odometria do robô.

Nesta etapa não são utilizados modelos bioinspirados devido ao custo computacional envolvido. Porém a abordagem possui a vantagem de não acrescentar as incertezas além da odometria durante o processo, uma vez que o tamanho do pacote de energia não é modificado.

4.1.4.3 Excitação Externa

A excitação externa é utilizada para criar uma ligação entre as *Place Cells* e *Local View Cells*. Essa ligação é importante devido às características de correção do erro de localização ser dependente de uma taxa proveniente da *Place Cell* e outra taxa das *Local View Cells*.

Essa ligação criada possuirá um peso sináptico calculado através de um aprendizado não supervisionado associativo. A cada instante que tanto a *Local View Cells* quanto a *Place Cell* estiverem ativas, será reforçada a ligação entre eles, com a característica dessa ligação não ser esquecida. Nesse contexto, o peso sináptico B é adquirido através da Equação 4.10

$$B_{ix',y',z'} = max(B_{ix',y',z'}, \lambda L_i r_{x',y',z'}), \tag{4.10}$$

onde, L_i é a taxa de ativação da *Local View Cell* i, $r_{x',y',z'}$ a taxa de ativação do neurônio x', y', z' e λ uma constante com valor definido para o aprendizado. Após essa etapa, é calculada a quantidade do valor da excitação externa de um neurônio da *Place Cell*:

$$\Delta r_{x',y',z'} = \sum_{i=1}^{m} B_{ix',y',z'} l_i. \tag{4.11}$$

4.1.4.4 Normalização

A normalização da rede neural é realizada com o intuito de limitar a atividade dos neurônios na rede, e assim evitando um crescimento ilimitado da rede que ocasionaria uma interferência no seu real comportamento.

O valor máximo da ativação de um neurônio é definido em uma unidade, e valor mínimo em zero. Primeiramente, como apresentado na Equação 4.12, é encontrado dentro da rede neural o neurônio com a maior ativação:

$$\eta = max(r_{x',y',z'}). \tag{4.12}$$

Após isso é realizada a divisão de todos os neurônios pelo valor do máximo, conforme apresentado na Equação 4.13:

$$r^n_{x',y',z'} = \begin{cases} \frac{r^n_{x',y',z'}}{n}, & r^n_{x',y',z'} > 0 \\ \\ 0 & r^n_{x',y',z'} \leq 0 \end{cases} \tag{4.13}$$

4.1.5 Mapa de Experiências

O mapa de experiência é responsável por criar um mapa semi-métrico, topológico e representativo do ambiente em tempo real, a partir de toda a movimentação do robô. Quando uma nova experiência ocorre, um novo nodo é adicionado ao mapa de experiência conectado com os seus vizinhos.

Um nodo de experiência (e_i) é uma tupla que armazena todas as informações que decorrem a partir da *Local View Cell* (L), a ativação da *Place Cell* (R) e a sua posição espacial no mapa de experiência (p_i) sendo demonstrado na Equação 4.14:

$$e_i = \{R^i, L^i, p_i\}. \tag{4.14}$$

A primeira posição do mapa de experiência por padrão é definida na posição $(0, 0, 0)$, e as demais são relativas a anterior.

4.1.5.1 Criação de Experiências

Conforme apresentado anteriormente, cada experiência é uma tupla de três elementos com os valores da *Local View Cell*, *Place Cell*, e posição no mapa de experiência. Quando ocorre a modificação visual ou de posição é criada uma nova experiência, a qual é comparada com as demais experiências do mapa para encontrar experiências similares.

A cada possível nova experiência é calculada a sua taxa de ativação em relação as outras experiências do mapa, conforme demonstrada na Equação 4.15:

$$S^i = \alpha_R|R^i - R| + \alpha_L|L^i - L|, \tag{4.15}$$

onde S^i é taxa de similaridade com outra experiência, R a taxa que representa a *Place cell*, L a taxa da *Local View Cell*, α_R e α_L são constantes que ponderam o peso de cada comparação. Caso a métrica S exceda um limiar pré-definido quando comparado com todas as outras experiências, será ativada a experiência que possuir maior taxa de similaridade. Senão, cria-se uma nova experiência no mapa a partir do deslocamento do robô e da experiência anterior como demonstrado na Equação 4.16:

$$e_j = \{R^j, L^j, p^i + \Delta p^{ij}\}, \tag{4.16}$$

onde e_j é a experiência atual, p^i a posição anterior no mapa e Δp^{ij} o deslocamento medido através da odometria. Assim, toda nova experiência do mapa estará ligada a anterior de forma topológica e com a informação de deslocamento armazenado através de δp^{ij}.

4.1.5.2 Fechamento de *loop*

O fechamento de *loop* ocorre quando um local previamente visitado pelo robô é reconhecido, ou seja, a taxa de similaridade calculada na Equação 4.16 possuir valor maior que o limiar selecionado.

A posição do robô é sensível aos erros de odometria. Assim, o fechamento de *loop* é responsável por reduzir o erro acumulado e auxiliar numa melhor estimativa de posição do robô.

A métrica proposta por Silveira (2015) para a correção do mapa utiliza a experiência reconhecida (e_k) da seguinte maneira:

1. Cria-se uma nova experiência no mapa, com as representações das *Local Views Cells* e *Place Cells* atuais.
$$e_j = \{R^j, L^j, p^i + \Delta p^{ij}\}, \tag{4.17}$$
onde i refere-se a experiência anterior e a j atual.

2. Cria-se a ligação entre as duas experiências.
$$t_{ij} = \{\Delta p^{ij}\}. \tag{4.18}$$

3. Cria-se outra ligação entre a experiência atual e a reconhecida, baseada nas informações visuais das duas imagens.
$$t_{jk} = \{\Delta p^{im}\}, \tag{4.19}$$

onde Δp^{im} é o deslocamento calculado através das duas imagens.

A criação de uma nova experiência vem da característica do ambiente subaquático, uma vez que o robô não está na mesma posição da aquisição das imagens. Após isto, é calculado o erro (Equação 4.20) entre a experiência atual e a experiência reconhecida e o deslocamento calculado através da imagem:

$$\epsilon = p_j + \Delta p^{im} - p_k. \tag{4.20}$$

O erro então é transferido para todas as experiências entre o intervalo de e_k e e_j, as ligações t são atualizadas pela Equação 4.21:

$$\Delta t = -\frac{\epsilon}{n+1}. \tag{4.21}$$

A partir disso, todas as posições são recalculadas através do peso das arestas. Em Zaffari et al. (2016) foi proposto o uso da equação de correção do mapa do algoritmo RatSLAM, estendendo a versão do 2D para o 3D. A equação leva em conta todos os movimentos registrado pelo robô e, após a detecção de *loop*, o mapa começa a ser corrigido conforme a Equação 4.22:

$$\Delta p_j = \alpha \left(\sum_{j=0}^{N_f} (p_k - p_j - \Delta p_{j,k}) - \sum_{l=1}^{N_t} (p_j - p_l - \Delta p_{l,j}) \right), \tag{4.22}$$

onde α é uma constante de correção do mapa, N_t é o número de conexões de outras experiências para a experiência atual (e_j), N_f o número de conexões da experiência atual para outras experiências, p_k a posição da experiência anterior, p_l a posição da próxima experiência e $\Delta p_{l,j}$ a distância percorrida pelo robô entre as experiências l e j.

Também em (ZAFFARI et al., 2016) é realizado um comparativo entre as duas equações de correção de mapa atualmente implementadas no algoritmo DolphinSLAM. A equação demonstrou uma diminuição maior no erro, também desenvolvendo um mapa final mais condizente com o *ground truth*. Duarte et al. (2016) apresentam *datasets* simulados para ambientes subaquáticos e um comparativo entre o algoritmo DolphinSLAM e o EKF, onde o dolphinSLAM encontrou resultados similares porém com custo computacional menor.

4.2 Modelo 2D CANN

O modelo da CANN proposto neste trabalho são baseadas nas células localizadas nos primatas e roedores, as quais são apresentados em Stringer et al. (2002a), Stringer et al. (2002b). Stringer et al. (2002a) apresenta o modelo de CANN para representar

um ambiente de duas dimensões, e em Stringer et al. (2002b) apresenta um modelo de comportamento das *head direction cells* as quais . Na Figura 29 apresenta-se o módulo que será modificado para acrescentar o comportamento da correnteza.

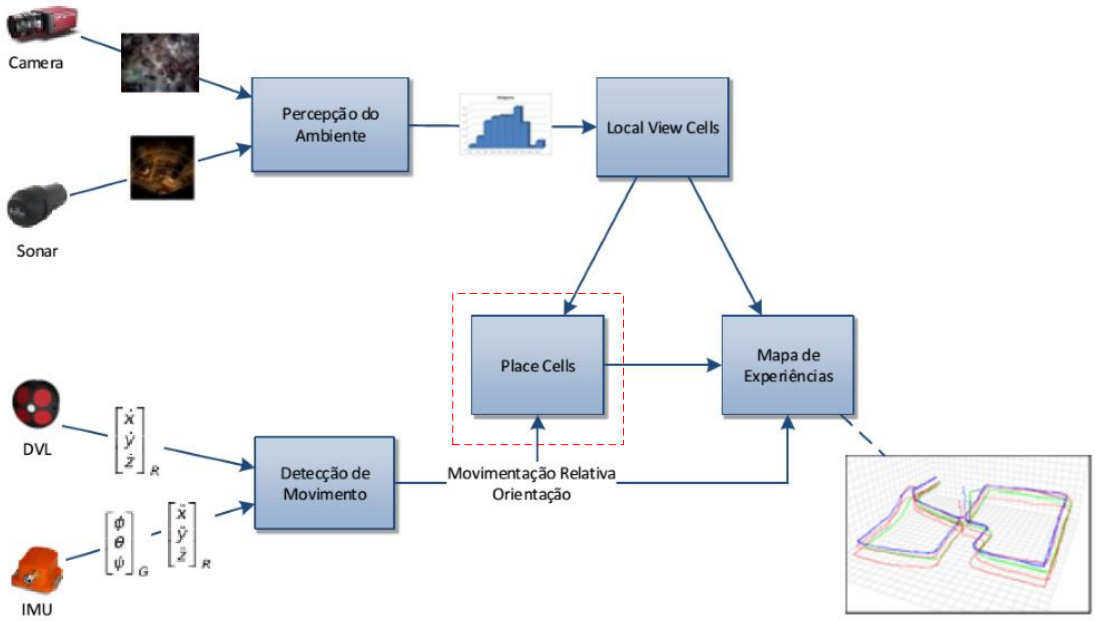


Figura 29 – Arquitetura do algoritmo DolphinSLAM. No quadrado vermelho apresenta-se o módulo a ser modificado.

A estabilização do modelo quando não há movimento é apresentado no trabalho (STRINGER et al., 2002a), onde a *place cell* mais ativa é sempre a mesma independente do número de iterações realizadas e a única forma de alterar a *place cell* mais ativa é através do reconhecimento de uma pista visual previamente visualizada. Na Figura 30 é apresentado um exemplo de ambiente e sua representação e localização através de uma rede de *place cell*.

Percebe-se que a rede representa um ambiente retangular onde o robô estará atuando. Os círculos modelam cada *place cell* do sistema. Cada *place cell* do modelo é responsável por descrever uma sub-área de atuação do robô no ambiente (x_i, y_i), ou seja, onde a ativação possuirá valor máximo. A cada iteração do algoritmo uma nova imagem é processada e é comparada com todas as imagens das outras *place cell*. Caso seja encontrada semelhança entre a entrada visual da rede e alguma *place cell*, ativa-se a *place cell* correspondente.

Na Equação 4.23 é apresentada a dinâmica de ativação cada *place cell* (h_i):

$$\tau \frac{\mathrm{d}h_i^p(t)}{\mathrm{d}t} = -h_i^p(t) + \frac{\phi_0}{C^P} \sum_j (w_{ij}^{RC} - w^{INH}) r_j^p(t) + I_i^V \tag{4.23}$$

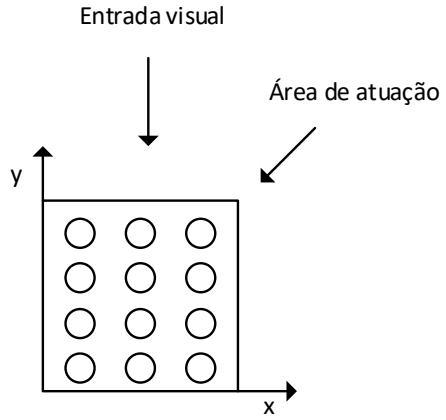


Figura 30 – Exemplo da área de atuação do robô.

O primeiro termo a direita da igualdade da Equação 4.23 ($-h_i^p(t)$) é o termo de decaimento da rede, onde diminui-se o valor da ativação calculada no instante anterior.

O segundo termo ($\frac{\phi_0}{C^P}\sum_j(w_{ij}^{RC} - w^{INH})r_j^p(t)$) apresenta os efeitos das ligações recorrentes da CANN, onde ϕ_0 é uma constante, C_P o total de ligações entre todos os neurônios ,w_{ij} é o peso sináptico entre a *place cell* j e a *place cell* i, e $r_j^p(t)$ é a taxa de ativação da *place cell* j.

O peso sináptico (w_{ij}) é calculado através da taxa de ativação da *place* j e da *place* i utilizando um aprendizado de *hebb*, como demonstrado na Equação 4.24:

$$w_{ij}^{RC} = k_1 r_i^p r_j^p. \tag{4.24}$$

onde k_1 é uma constante de treinamento, as taxas de ativação de cada *place cell* possuem um padrão gaussiano e são calculadas através da distância entre a *place cell* atual e a *place cell* mais ativa da rede, como demonstrado na Equação 4.25

$$r_i^p = exp[\frac{-(\sqrt{(x_i - x)^2 + (y_i - y)^2})}{2(\sigma^p)^2}] \tag{4.25}$$

onde σ_p é o desvio padrão da função, x_i e y_i são as coordenadas atuais da *place cell* e x e y as coordenadas do neurônio mais ativo.

O terceiro termo (I_i^V) é a entrada visual da rede. Uma vez que um local previamente visitado for reconhecido a *place cell* correspondente é ativada com um padrão gaussiano utilizando a Equação 4.24 sem a influência da constante de treinamento. Caso não seja reconhecida nenhuma pista visual do ambiente, o valor de I_i^V é definido como zero.

Na ausência das pistas visuais as ligações idiotéticas são utilizadas para descrever o movimento do robô e consequentemente responsáveis por atualizar a ativação das *place cells* na sua direção. No modelo, existem duas possíveis entradas idiotéticas: *head direction cells* e *forward velocity cells*.

As *head direction cells* representam a orientação do robô no ambiente e as ligações sinápticas estão organizadas conforme Stringer et al. (2002b). As *forward velocity cells* são utilizadas para representar o movimento do agente, onde a taxa de ativação aumenta conforme a velocidade do agente.

A partir da Equação 4.23 adiciona-se o comportamento das ligações idiotéticas e assim, calcula-se a ativação de cada *place cell* (h_i) do modelo através da Equação 4.26:

$$\tau \frac{\mathrm{d}h_i^p(t)}{\mathrm{d}t} = -h_i^p(t) + \frac{\phi_0}{C^P} \sum_j (w_{ij}^{RC} - w^{INH}) r_j^p(t) + I_i^V + \frac{\phi_1}{C^{P \times HD \times FV}} \sum_{j,k,l} w_{ijkl}^{WC} r_j^p r_k^{HD} r_l^{FV}. \tag{4.26}$$

O termo $(\frac{\phi_1}{C^{P \times HD \times FV}} \sum_{j,k,l} w_{ijkl}^{WC} r_j^p r_k^{HD} r_l^{FV})$ representa o efeito das ligações idiotéticas implementadas, onde ϕ_1 é uma constante $C^{P \times HD \times FV}$ é a combinação de todas ligações idiotéticas entre todos os neurônios presentes no modelo, r_j^p é a taxa de ativação da *place cell* j, r_l^{FV} é o neurônio responsável por incrementar a taxa de disparo da *forward velocity cell* e r_k^{HD} os neurônios do tipo *head direction cells* do robô calculados por Equação 4.27 e Equação 4.28 apresentadas por Stringer et al. (2002b), onde x e x_i é a diferença em graus do neurônio mais ativo para o atual:

$$s_i^{HD} = MIN(|x_i - x|, 360 - |x_i - x|), \tag{4.27}$$

$$r_i^{HD} = exp[\frac{-(s_i^{wh})^2}{2(\sigma^{wh})^2}], \tag{4.28}$$

w_{ijkl} corresponde as forças das conexões efetivas e calcula-se através da Equação 4.29:

$$w_{ijkl} = k_2 r_i^p r_j^{-p} r_l^{FV} r_k^{HD}, \tag{4.29}$$

onde k_2 é uma constante, r_i^p é a taxa instantânea da *place cell* i, r_k^{HD} é taxa de ativação da *head direction cells* do robô, r_l^{FV} a taxa do neurônio da *forward velocity cells*, e r_j^{-p} é calculado através da *trace rule* e demonstrado na Equação 4.30:

$$r_j^{-p} = (1 - \eta) r_j^p(t + \delta t) + \eta r_j^{-p}(t) \tag{4.30}$$

η é responsável por determinar a contribuição da taxa de disparo da *trace rule*.

Na Figura 31 é apresentado a arquitetura da rede neural para o modelo de *place cell 2D*:

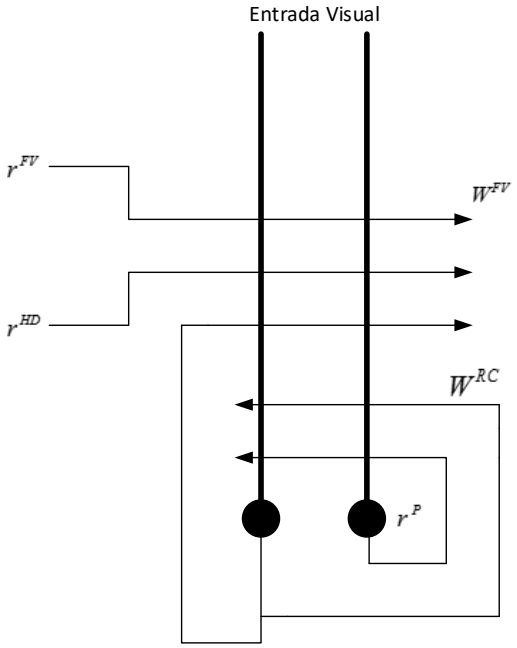


Figura 31 – Arquitetura da rede para o modelo de *place cell* 2D.

A rede neural recebe entradas exteriores de três diferentes formas: Entrada visual, *head direction cells* e *forward velocity cells*. Os pesos recorrentes entre as *place cells* são apresentados pelo W^{RC} e os pesos idiotéticos pelo W^{FV}. Através deste modelo de CANN $2D$, serão adicionados neurônios específicos para acrescentar o comportamento da correnteza na suas ligações idiotéticas.

No próximo capítulo será apresentado o modelo proposto neste trabalho. Será integrado ao algoritmo DolphinSLAM a informação proveniente da correnteza marítima, para isso, será necessário modificar a CANN atualmente implementada.

5 Modelo Proposto

Neste capítulo será apresentada uma proposta de utilização de uma rede CANN para a modelagem do efeito da correnteza no deslocamento, localização e mapeamento de um robô subaquático. Primeiramente é contextualizado o problema ocorrido através da correnteza e, posteriormente o sistema proposto. Por último, apresenta-se o treinamento da rede, mostrando a estabilidade do pacote de energia da rede e a característica das ligações recorrentes e idiotéticas.

5.1 Efeito das Correntes Marítimas através de uma CANN

Segundo Pedlosky (2013), os oceanos são fluídos rotativos e estratificados, onde os movimentos verticais são praticamente insignificantes quando comparado com os movimentos horizontais.

A navegação autônoma neste tipo de ambiente é considerada um dos principais desafios para robótica móvel. As características do meio subaquático associadas também a presença de água e de partículas em suspensão trazem uma série de inconvenientes relacionados à percepção e atuação do robô durante a sua navegação. Deste podemos citar:

- Os efeitos de dispersão, atenuação e reflexo dos sinais luminosos, acústicos e eletromagnéticos utilizados por diferentes sensores perceptivos tanto ativos como passivos.
- A influência da correnteza nas operações utilizando o robô neste ambiente. Por exemplo no seu deslocamento e/ou ações de *pick/place* associados à manipulação do objeto.
- As características naturais do ambiente, onde estruturas bem conhecidas, desenvolvidas pelo homem, como vias, corredores, paredes, não estão presentes.
- A monotocidade e ausência de bordas e marcas do meio provocada pelo desgaste das superfícies em decorrência da erosão, bem como a predominância cromática da cor da água presente.
- A difícil modelagem da dinâmica da cena que pode decorrer da presença esporádica e imprevista de cardumes.

As correntes marítimas são responsáveis por afetar a trajetória realizada pelo robô e pelo consumo de energia (LIU et al., 2013). A trajetória é afetada através do arraste

sofrido pelo robô na direção da correnteza. O consumo de energia possui a tendência de aumentar, uma vez que se o robô estiver atuando contra o sentido da correnteza, utilizará mais energia para seguir o caminho estipulado. Assim, faz-se necessário o uso de modelos da correnteza para otimizar a trajetória realizada, o consumo de energia e consequentemente aumentando o tempo de operação do AUV.

Alguns trabalhos (STANWAY, 2010; MEDAGODA et al., 2011; FIRING, 1991) exploram o uso dos dados da correnteza provenientes de um ADCP para auxiliar na navegação.

Nesse contexto, é proposto alterar o modelo atual da CANN de forma a possibilitar a interpretação, modelagem e tratamento das informações provenientes das correntes marítimas, e assim, obtendo uma representatividade mais real do ambiente e consequentemente melhorando a abordagem para a resolução do problema do SLAM. Na próxima seção é apresentado o modelo proposto trabalho.

5.1.1 Modelo para a adição da correnteza

Na adição da correnteza no modelo apresentado na seção 4.2, as parcelas referentes as ligações recorrentes da *place cell*, entrada visual e o termo de decaimento não serão alterados conforme demonstrado na Equação 4.23. Entretanto propõe-se a modificação do termo referente as ligações idiotéticas.

As ligações idiotéticas referem-se a capacidade de deslocamento da atividade da rede neural durante a ausência de reconhecimento da informação visual do ambiente. Devido à esta capacidade, propõe-se o acréscimo do comportamento da correnteza através do uso da sua orientação e velocidade durante as etapas de treinamento e execução do modelo. Assim, encontrando uma representação mais real da posição em que o robô encontra-se no ambiente.

Parte-se da hipótese que a presença da correnteza e seu efeito no deslocamento do robô, possam ser incorporados diretamente na dinâmica da ativação dos pacotes de ativação da CANN. Camadas de neurônios idiotéticos representariam as forças provenientes da corrente. O sistema aprenderia a influência de diferentes padrões de correnteza nos novos posicionamentos do robô, a partir da variação das conexões entre a camada idiotética e as *place cells* do robô.

A proposta é que durante a etapa de aprendizado dos pesos sinápticos, cada *place cell* é representada pela ativação da própria *place cell*, com a *place cell* recentemente mais ativa e com a entrada das ativações dos neurônios que compõe a parte idiotética da rede. O resultado deste aprendizado é que no momento em que não houver reconhecimento de pistas visuais, este comportamento previamente treinado será responsável por mover o pacote de energia da *place cell* atual na direção da *place cell* que foi ativada

subsequentemente durante o aprendizado.

Para a proposta do modelo adiciona-se a informação da correnteza do modelo através dos neurônios do tipo *head direction cells* e *forward velocity cells*. O primeiro responsável por indicar a orientação em que a correnteza está e o segundo a sua velocidade. A arquitetura da rede neural e as suas ligações são apresentados na Figura 32:

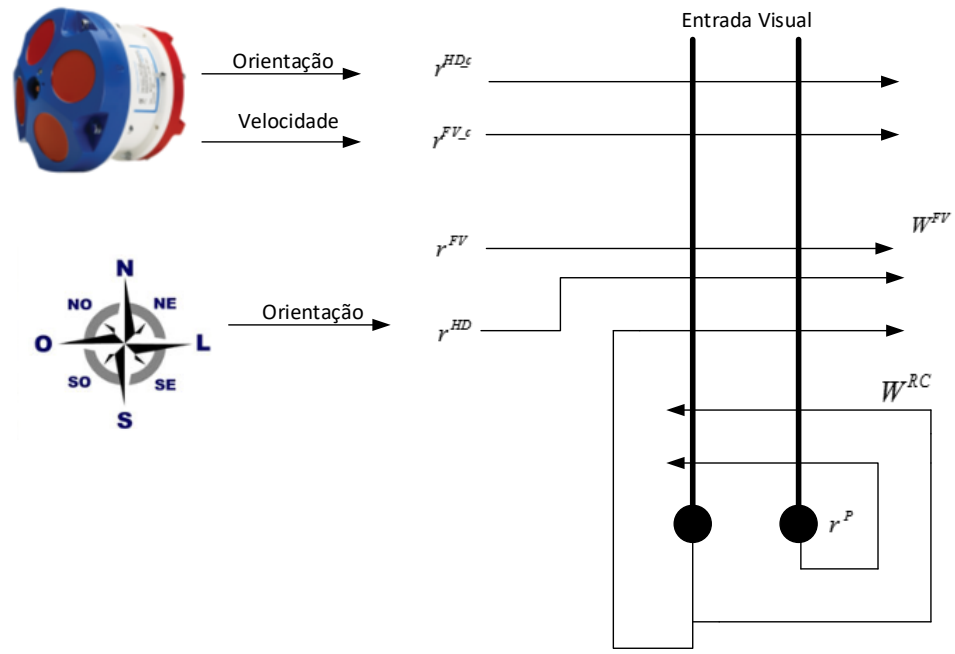


Figura 32 – Arquitetura da rede para o modelo da *place cell* 2D com a adição da informação da correnteza.

A partir da Equação 4.23 com o acréscimo da informação das ligações idiotéticas, calcula-se a ativação de cada *place cell* (h_i) através da Equação 5.1:

$$\tau \frac{\mathrm{d}h_i^p(t)}{\mathrm{d}t} = -h_i^p(t) + \frac{\phi_0}{C^P} \sum_j (w_{ij}^{RC} - w^{INH}) r_j^p(t) + I_i^V + \frac{\phi_1}{C^{P \times WCh \times WCv}} \sum_{j,k,l,f,g} w_{ijklfg}^{WC} r_j^p r_k^{WC} r_l^{WC} r_f r_g. \tag{5.1}$$

A fase de treinamento do modelo para a definição dos pesos sinápticos da parte idiotética envolve todas as *place cells*, a *head direction cells* e *forward velocity cells* do robô, e as *head direction cells* e *forward velocity cells* da correnteza. Durante a o treinamento os pesos sinápticos são atualizados conforme a Equação 5.2

$$\delta w_{ijklfg} = k_2 r_i^p r_j^{-p} r_f r_g r_k^{WC} r_l^{WC}, \tag{5.2}$$

onde δw_{ijklfg} é a mudança dos pesos sinápticos, r_i^p a taxa de ativação da *place cell* i, r_j^{-p} é o valor correspondente da trajetória e calculado através da Equação 4.30, r_f a taxa de *head direction cell* do robô, r_g a taxa da *forward velocity cell* do robô, r_k^{WC} a *head direction cell* da correnteza, r_l^{WC} *forward velocity cell* da correnteza e k_2 a taxa de aprendizado.

5.2 Treinamento do modelo

De forma a melhor elucidar o modelo proposto, apresenta-se nesta seção um exemplo do treinamento da rede CANN proposta. O treinamento dos pesos da rede é uma etapa anterior a execução do modelo proposto, no qual são calculados todos os possíveis pesos para as ligações entre os neurônios da rede. No modelo apresentado na subseção 5.1.1 utiliza-se quatro tipos de entrada para atualizar a ativação das *place cells*:

- Entrada visual utilizada durante o aprendizado.
- Entrada referente a orientação do robô.
- Entrada referente a orientação da correnteza.
- Entrada referente a velocidade da correnteza.

As entradas visuais para do modelo são obtidos através de câmera ou SONAR as quais são responsáveis por descrever o ambiente. A orientação da correnteza e velocidade são obtidos através do uso de um ADCP. E a orientação da do robô é obtido através do uso de sensores específicos de orientação, como a bússola.

Para o exemplo de treinamento do modelo assume-se que o espaço bidimensional será discretizado em 2500 células espaciais distribuídas em um grid de 50×50, 8 neurônios para a orientação do robô, 8 neurônios para orientação da correnteza, e 1 neurônio para a velocidade da correnteza.

A primeira etapa realizada pelo algoritmo refere-se ao treinamento das ligações recorrentes entre todas as *place cells* da rede. Durante esta etapa o robô move-se através de todo ambiente com as pistas visuais disponíveis, ativando cada *place cell* ao máximo e assim, calculando o peso sináptico entre todos os neurônio da rede. Os pesos sinápticos recorrentes são definidos inicialmente como zero e esta etapa é repetida 100 vezes para gerar os pesos das ligações recorrentes. O treinamento é importante para que na ausência das pistas visuais e das ligações idiotéticas, o robô consiga manter a sua localização na mesma *place cell*.

As conexões recorrentes (w_{ij}^{RC}) entre cada *place cell* são calculadas utilizando a Equação 4.24. A soma dos pesos da ligação de um neurônio para todos os outros da rede

é responsável por definir a sua contribuição. Na Figura 33 apresenta-se o comportamento da rede quando o neurônio central da rede é o mais ativo.

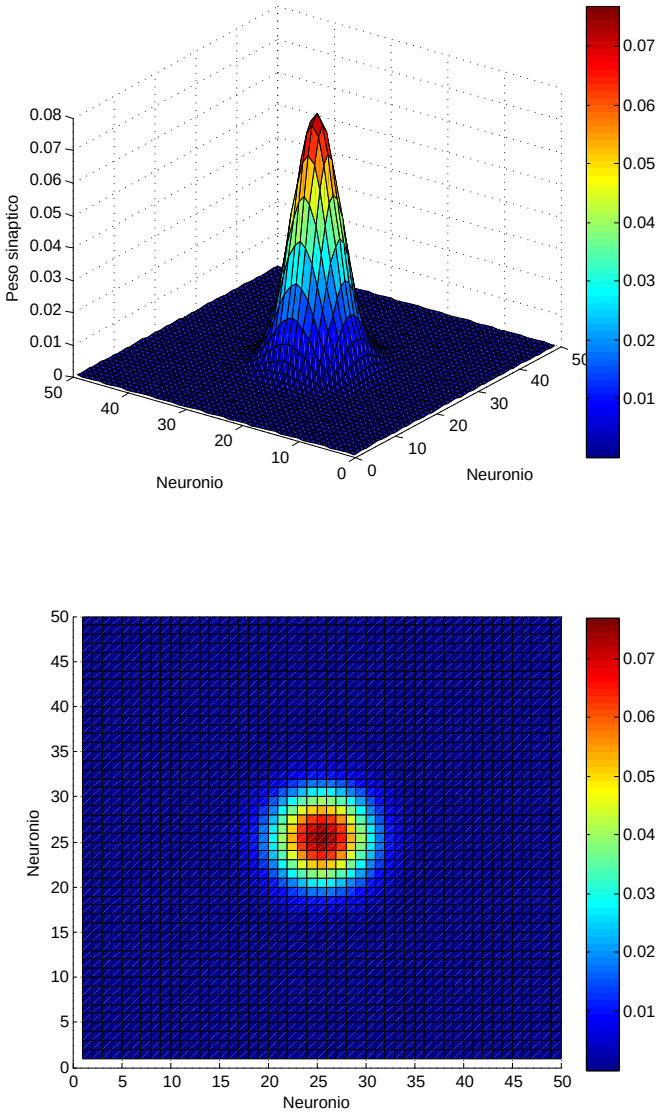


Figura 33 – Pacote de ativação das ligações recorrentes sem a adição da informação das ligações idiotéticas

Na Figura 33 é possível perceber que os neurônios próximos ao de maior ativação possuem ligações recorrentes fortes quando comparados com os mais distantes. Outra característica encontrada é a simetria dos pesos recorrente em relação ao nodo central o que é importante para manter a estabilidade da rede quando o robô encontra-se parado.

Após o treinamento das ligações recorrentes é realizado o treinamento das ligações idiotéticas (w_{ijklf}) da CANN. A proposta deste tipo de ligação é que na ausência de correspondência entre os locais previamente visitados, o pacote de energia desloca-se a partir do comportamento previamente treinado. No caso proposto neste trabalho, na

direção em que a correnteza está afetando o veículo.

O treinamento das ligações idiotéticas envolve o cálculo de todos os pesos sinápticos (w_{ijklf}) utilizando a combinação das *place cells* i e j, das *head direction cells* e *forward velocity cells*. Nessa etapa, o robô se move por todo o ambiente, calculando os pesos conforme a *place cell* que está mais ativa, a orientação do robô, e a orientação e velocidade da correnteza. Nesta etapa os pesos sinápticos iniciais são definidos como zero, e a etapa de treinamento é repetida por 100 vezes para calcular os pesos sinápticos das ligações idiotéticas.

Para apresentar o perfil dos pesos sinápticos aprendidos pela rede é definido o neurônio central da rede como a *place cell* mais ativa e apresentando os pesos dos neurônios com variação no eixo Y. Essa variação é demonstrada pela linha pontilhada na Figura 34.

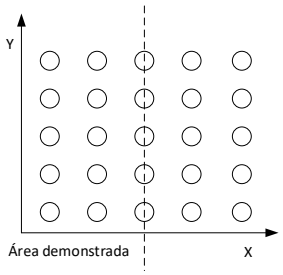


Figura 34 – Perfil dos pesos das conexões recorrentes e sinápticos.

Na Figura 35 é apresentado o comportamento dos pesos sinápticos da rede no intervalo pontilhado após o treinamento das ligações idiotéticas. É realizado um comparativo entre os pesos das ligações recorrentes e idiotéticas, utilizando a orientação da correnteza para o norte.

No gráfico percebe-se que as ligações recorrentes possuem maior peso sináptico que as ligações idiotéticas. Também nas ligações recorrentes (w_{ij}) o peso é maior no neurônio central enquanto que nas ligações idiotéticas (w_{ijklf}) o peso maior encontra-se deslocado do neurônio central e também possui um comportamento assimétrico. O comportamento assimétrico e o peso sináptico maior estar deslocado do neurônio central é importante para o modelo, uma vez que isso permitirá o deslocamento da atividade da rede neural no sentido da correnteza treinada neste modelo.

Na Figura 36 é apresentado o comportamento dos pesos das ligações idiotéticas quando a correnteza encontra-se para as outras direções, para tal assumimos as quatro principais direções: Norte, Sul, Leste e Oeste.

Como apresentado na Figura 36, os pesos para as direções oeste e leste são iguais uma vez que o movimento esperado é em relação ao eixo X e a área plotada possui variação no eixo Y. Os pesos para a direção norte e sul são assimétricos onde o peso mais ativo para

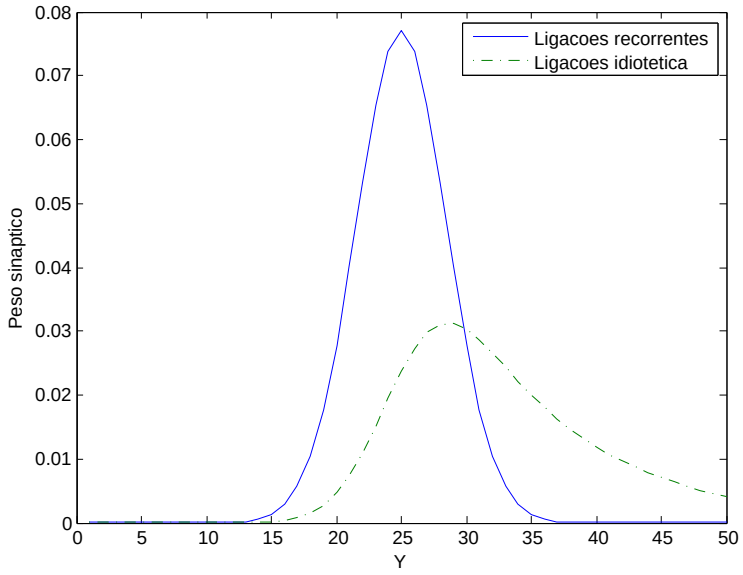


Figura 35 – Perfil dos pesos das conexões recorrentes e idiotéticas.

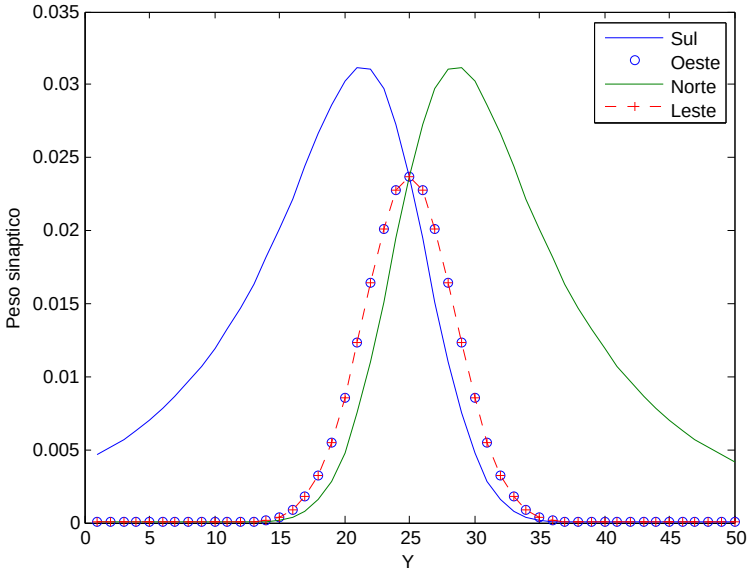


Figura 36 – Perfil dos pesos das conexões idiotéticas nas diferentes orientações.

a região norte encontra-se nos neurônios ao norte do neurônio central, enquanto para a direção sul o neurônio encontra-se ao sul do neurônio central. A orientação da correnteza com a maior taxa de ativação será responsável por determinar a direção em que o robô será deslocado.

Na Figura 37 são apresentados os comportamentos dos pesos das ligações conforme a orientação em que a correnteza está afetando o robô.

No quadrante superior da esquerda apresenta-se os pesos para a correnteza com a direção norte, no quadrante inferior da esquerda para o sul. Os dois pesos possuem característica assimétrica porém com a direção correspondente. O gráfico superior da direita representa a direção oeste e o da inferior a direita a direção leste. Devido à área plotada possuir variação no eixo Y os pesos apresentados possuem um comportamento simétrico. Porém, se modificar à área plotada para a variação no eixo X, o comportamento dos pesos da direção oeste e leste serão assimétricos com a maior ativação na direção das mesmas.

Neste capítulo foram apresentados os efeitos que a correnteza influi no deslocamento de um robô no ambiente subaquático. Também apresentou-se a modificação no modelo bioinspirado para a CANN, com os tipos de neurônios utilizados no experimento e o equacionamento que os regem e posteriormente o treinamento utilizado. No próximo capítulo serão apresentados os resultados encontrados utilizando o modelo proposto.

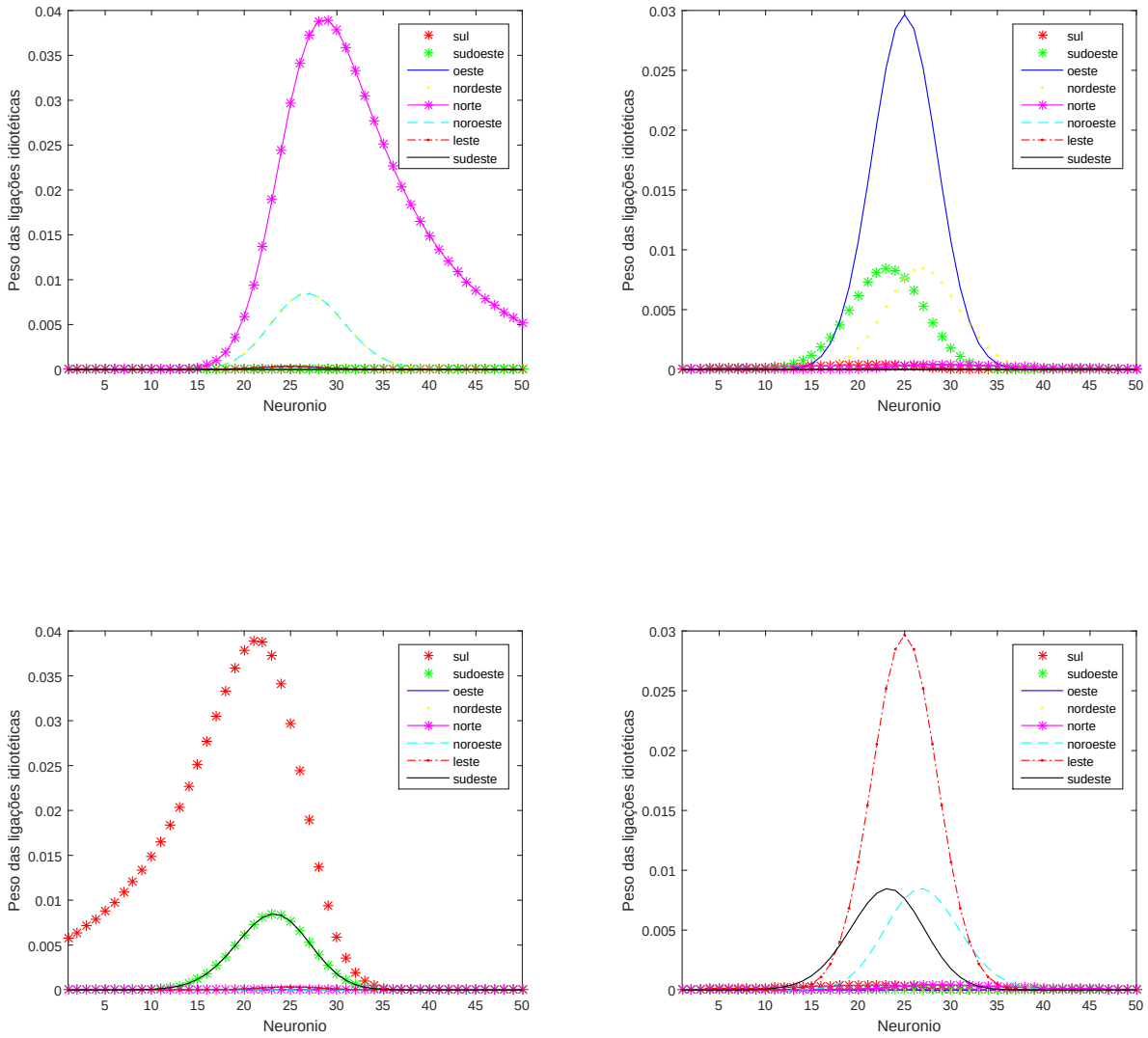


Figura 37 – Exemplo dos pesos sinápticos das ligações idiotéticas para a correnteza nas diferentes direções.

6 Experimentos

Neste capítulo serão apresentados os diferentes tipos de experimentos utilizados para a validação da abordagem proposta. Nas etapas de treinamento realizadas pelo modelo proposto, utilizou-se os dados da correnteza fornecidas pelo simulador utilizado, para a aplicação em *datasets* reais, a orientação e velocidade da correnteza será fornecida através de um ADCP. Outra característica utilizado em todos os experimentos, é o uso da correnteza com uma velocidade e direção constante.

No primeiro cenário utilizado o robô não exerce movimento, sendo só a correnteza responsável pelo seu deslocamento. No segundo cenário, o robô exerce movimento em um ambiente simulado utilizando o simulador gazebo. Por último, o modelo é avaliado em uma trajetória mais extensa e utilizando o cenário do segundo experimento.

6.1 Caso 1: Trajetória estacionária

Para o primeiro caso de estudo do modelo proposto utiliza-se *UUV Simulator* (MANHÃES et al., 2016). O *UUV Simulator* é um conjunto de *plug-ins* para ambientes subaquáticos no simulador gazebo, onde permite simular diferentes veículos, sensores como IMU, DVL, SBL, adicionar o comportamento da correnteza e modificar o ambiente simulado. Outra característica importante é a integração com o ROS, facilitando a aquisição de dados para os datasets utilizados durante a simulação.

No teste proposto, no robô não será realizado movimento através dos *thrusters* sendo somente a correnteza responsável pelo seu deslocamento no ambiente. Para este experimento, foi utilizado o robô RexROV equipado com câmera, IMU, DVL, SBL. O modelo do oceano utilizado é fornecido pelo simulador. Durante o experimento, o robô é deslocado por aproximadamente 30 metros.

A imagem representativa do robô e o oceano utilizado para gerar datasets é apresentado na Figura 38:

Na Figura 39 apresenta-se exemplos de algumas imagens capturadas pelo robô.

6.1.1 Parâmetros do Modelo

Na Tabela 2 são apresentados os parâmetros utilizados na configuração do modelo.

O número de neurônios do tipo *place cell* utilizados pelo modelo deve ser definido primeiramente em conjunto com a representação de ambiente que cada neurônio é responsável. O desvio padrão da gaussiana utilizada e as constantes de treinamento são

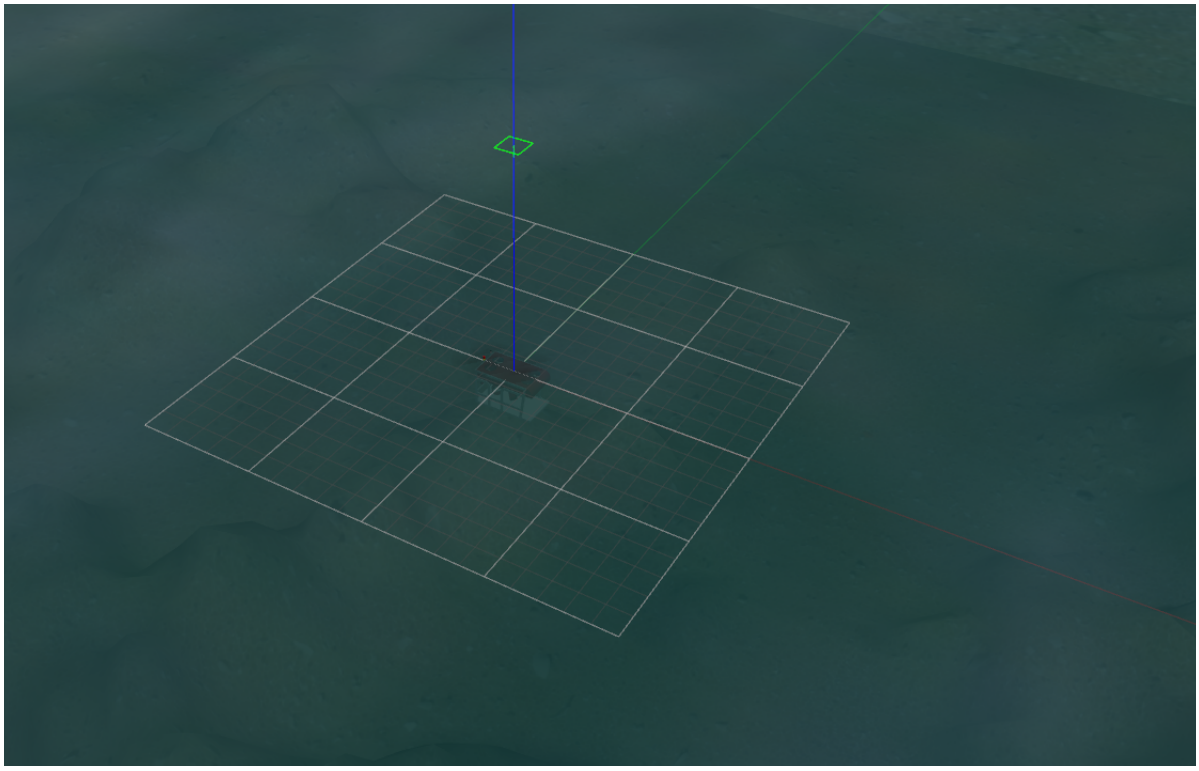


Figura 38 – Ambiente utilizado para a simulação da correnteza.

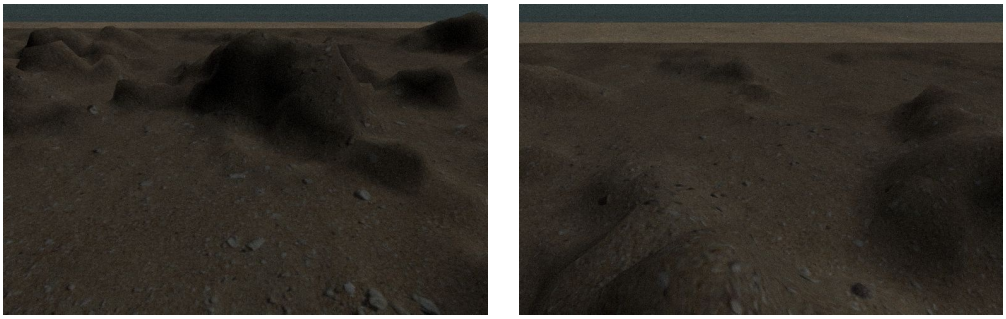


Figura 39 – Imagens coletadas durante o experimento no dataset simulado

importantes para descrever a contribuição dos pesos das ligações recorrentes e idiotéticas para cada neurônio do modelo.

Para as *head direction cells* configura-se três parâmetros: Número de neurônios, desvio padrão da gaussiana e a sua resolução. O número de neurônios é responsável por descrever a resolução da orientação que cada neurônio possuirá, uma vez que a orientação possuí 360◦ que são divididos pelo número de neurônios do tipo *head direction*. Devido à utilização de uma velocidade da correnteza constante no modelo, utiliza-se somente um neurônio para descrever.

Para as pistas visuais do modelo, utiliza-se o descritor SURF definindo o *threshold* para as *features*. Também é necessário definir o número de grupos que serão treinados

Tabela 2 – Parâmetros utilizados no modelo

Place Cell	
Neurônio	100×100
Distância entre os Neurônios	$0.3\ m$
σ_p	3.5
k_1	0.001
k_2	0.001
τ	1
η	0.9
ϕ_0	50000
ϕ_1	100000
w_{inh}	0.1
C_0	$\phi_0/Neurônio$
C_1	$\phi_1/160000$
Head Direction Cell	
Neurônio	8
σ_{hd}	20
Resolução	$45\ \circ$
Pistas visuais	
Threshold	150
Número de grupos BOW	700

utilizando BOW.

6.1.2 Resultados

Para o primeiro experimento, o modelo é treinado para a correnteza com velocidade $0.01\ m/s$ na direção nordeste, ou seja, com variação significativa nos eixos X e Y. Na Figura 40 são apresentados os neurônios mais ativos durante cada iteração do algoritmo.

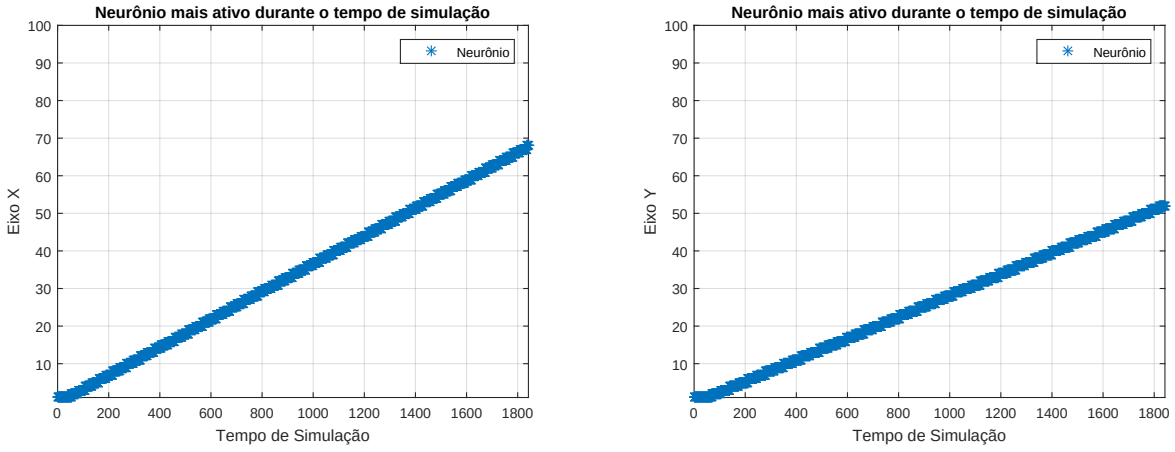


Figura 40 – Ativação dos neurônios durante o tempo de simulação.

Na Figura 40, a imagem à esquerda representa a posição da *place cell* no eixo X e a imagem a direita a posição no eixo Y. As ativações possuíram o comportamento esperado, onde ocorreu a variação das ativações dos neurônio em ambos os eixos.

A partir da localização (x_i,y_i) dos neurônios mais ativos durante a simulação, é possível criar um mapa representativo do ambiente. Na Figura 41 apresenta-se o mapa criado utilizando o modelo, o *ground truth* e o *dead reckoning*.

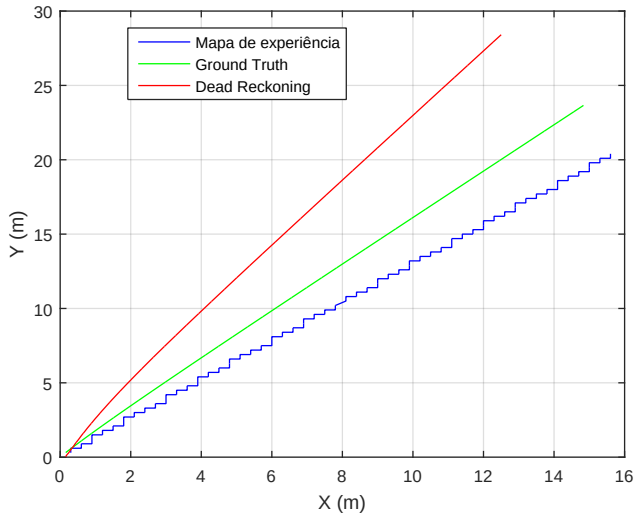


Figura 41 – Trajetória calculada durante o experimento simulado. Em verde a trajetória real do robô, em vermelho a trajetória criada através da odometria do robô, e em azul o mapa criado utilizando o modelo proposto.

Na Figura 41 é possível perceber que a trajetória calculada através do modelo possui um comportamento similar quando comparado com o *ground truth*. Neste experimento não ocorreu nenhum reconhecimento de pistas visuais uma vez que não ocorreu a passagem por um lugar previamente visitado devido ao movimento realizado pelo robô.

Na Figura 42 é apresentado o erro correspondente do modelo e *dead reckoning* em relação ao *ground truth*.

O erro calculado pelo modelo ficou menor na maior parte da simulação mesmo na ausência do reconhecimento das pistas visuais, possuindo alguns picos maiores no começo da simulação.

6.2 Caso 2: Ambiente Simulado

No segundo experimento é apresentado uma simulação utilizando o software Gazebo, porém, diferente do experimento anterior o robô realizará uma trajetória no ambiente utilizando os *thrusters* e também sob a influência da correnteza.

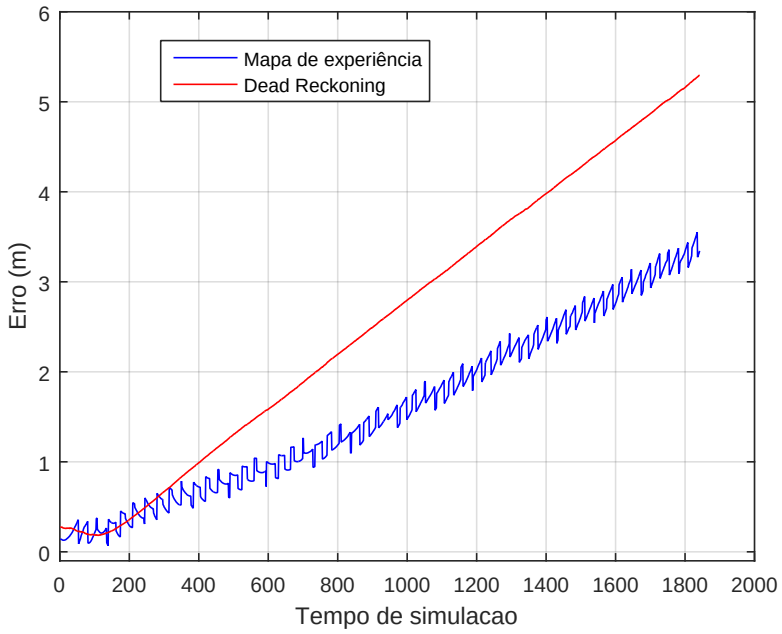


Figura 42 – Erro calculado entre o mapa de experiência e *Dead Reckoning* em relação ao *Ground Truth*.

Neste experimento também utiliza-se o robô RexROV equipado com os sensores IMU, DVL, SBL e câmera. O cenário utilizado foi desenvolvido pelo grupo de pesquisa, onde simula-se uma marina com 4 píer e alguns barcos estacionados no píer. Devido à similaridade do fundo do oceano utilizado pelo simulador, tornou-se necessário adicionar alguns elementos no fundo do oceano. Para isso, neste dataset foram adicionados objetos provenientes da intervenção humana como botas, barris e barcos afundados. Na Figura 43 apresenta-se o cenário desenvolvido para este experimento:

Na Figura 44 são apresentados exemplos de imagens capturados pelo robô durante o experimento, onde é possível observar alguns componentes adicionados no fundo do mar e as vigas de sustentação do píer.

6.2.1 Parâmetros do Modelo

Na Tabela 3 são apresentados os parâmetros utilizados na configuração do modelo.

6.2.2 Resultados

No segundo experimento, o modelo novamente é treinado para a correnteza com velocidade 0.02 m/s com a direção em leste. Entretanto nesta etapa é avaliado o comportamento do modelo durante a realização de uma trajetória utilizando os *thrusters* do robô. A trajetória realizada pelo robô é uma linha reta mantendo a direção do movimento

Figura 43 – Ambiente simulado do Yacht Clube Rio Grande para experimentos.

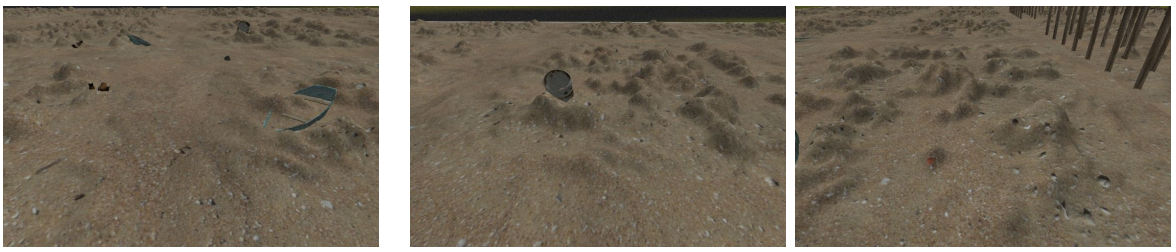


Figura 44 – Imagens coletadas durante o experimento no dataset simulado.

para o norte, ou seja, com o deslocamento no eixo Y. O caminho percorrido pelo robô durante a simulação é de aproximadamente 30 metros de operação.

Na Figura 45 é apresentado o mapa criado a partir da ativação dos neurônios.

Na Figura 45 foi apresentado o mapa criado utilizando a trajetória proposta no ambiente sobre a influência da correnteza. No mapa é possível notar que o modelo proposto encontrou um mapa condizente quando comparado ao *ground truth*. O mapa mostra que o deslocamento ocasionado pela correnteza não foi adicionado a odometria do simulador. Mesmo com este comportamento da odometria, o modelo proposto conseguiu apresentar uma trajetória mais parecida com o *ground truth*. Devido ao tempo de simulação, percebe-se que o mapa criado vai de encontro ao *ground-truth*, porém, percebe-se que utilizando um *dataset* com maior tempo de simulação o mapa de experiência criado, começaria a divergir com um maior erro.

Na Figura 46 são apresentadas as ativações dos neurônios no eixo X e Y em relação ao tempo de simulação.

A ativação dos neurônios em X e Y seguiram o comportamento do movimento

Tabela 3 – Parâmetros utilizados no modelo

Place Cell	
Neurônio	100×100
σ_p	3.5
k_1	0.001
k_2	0.001
τ	1
η	0.9
ϕ_0	50000
ϕ_1	100000
w_{inh}	0.1
C_0	$\phi_0/Neurônio$
C_1	$\phi_1/160000$
Head Direction Cell	
Neurônio	8
σ_{hd}	20
Resolução	45 ∘
Pistas visuais	
Threshold	150
Número de grupos BOW	700

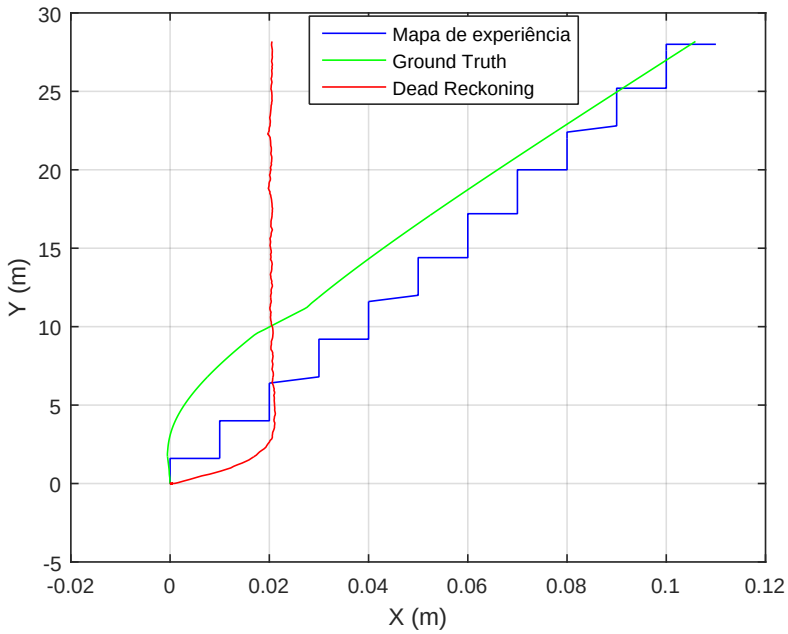


Figura 45 – Trajetória calculada durante o experimento simulado. Em verde a trajetória real do robô, em vermelho a trajetória criada através da odometria do robô, e em azul o mapa criado utilizando o modelo proposto.

do robô no ambiente. A mudança da ativação no eixo Y ser maior é esperado devido ao movimento proposto do robô. A mudança da ativação dos neurônios no eixo X refere-

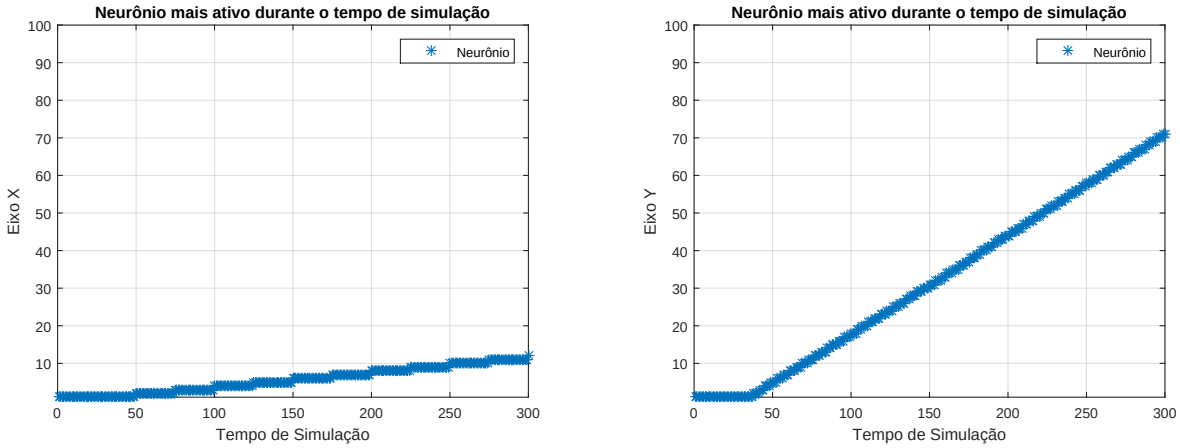


Figura 46 – Ativação dos neurônios durante o tempo de simulação.

se à influência da correnteza no movimento do robô. Para a ativação no eixo Y os 100 neurônios utilizados foram configurados para uma distância de 0.4 m, e para o eixo X cada neurônio 0.05 m.

Na Figura 47 é apresentado a relação do erro de posição do *dead reckoning* e do mapa criado em relação ao *ground truth*.

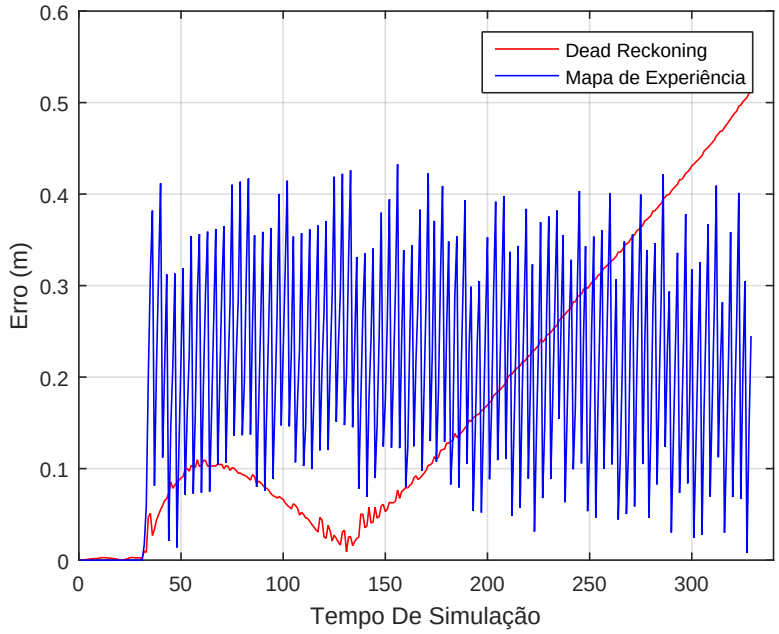


Figura 47 – Erro calculado entre o mapa de experiência e *Dead Reckoning* em relação ao *Ground Truth*.

O erro encontrado pelo modelo foi maior durante quase todo o experimento do que

o *dead reckoning*, porém o mapa criado a partir da ativação dos neurônios apresenta uma similaridade maior do que o *ground truth*. Na Figura 48 são apresentados os erros para o eixo X e eixo Y.

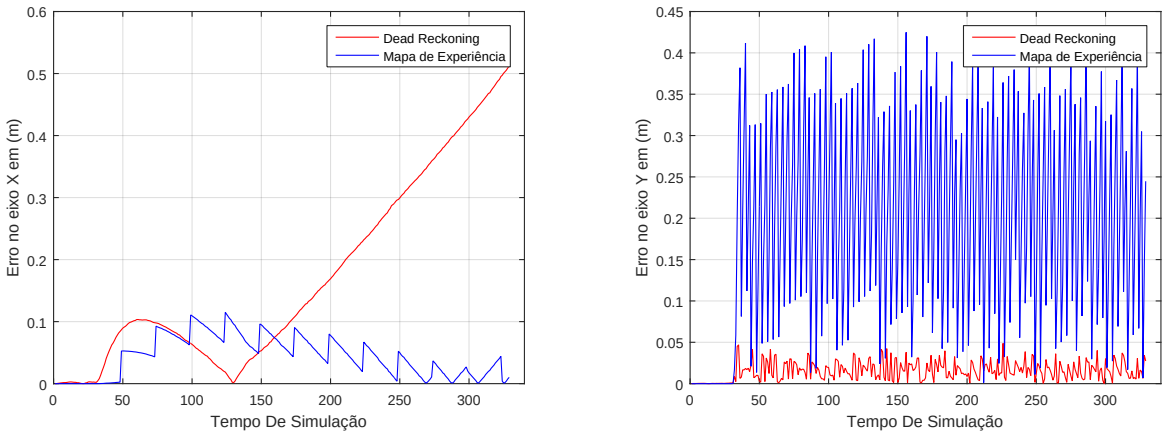


Figura 48 – Erro relativo ao eixo X e ao eixo Y.

É possível notar que o erro encontrado pelo modelo na direção da correnteza para o eixo X é menor devido à ação do modelo proposto.

6.3 Caso 3: Ambiente Simulado, Trajetória com voltas

No terceiro experimento novamente utilizou-se o software Gazebo e o cenário apresentado na seção 6.2. O robô nessa proposta também realizará uma trajetória utilizando os *thrusters* em um ambiente com a influência da correnteza.

6.3.1 Parâmetros do Modelo

Na Tabela 4 são apresentados os parâmetros utilizados na configuração do modelo.

6.3.2 Resultados

Para o terceiro experimento utilizou-se o mesmo cenário do segundo experimento e o modelo foi treinado para a correnteza com velocidade 0.03 m/s com a direção para leste. A trajetória proposta neste experimento é maior com aproximadamente 350 metros de deslocamento. O aumento da trajetória utilizando o mesmo cenário é proposto para apresentar a robustez do modelo para longas trajetórias. Na Figura 49 é apresentado o mapa criado a partir da ativação dos neurônios.

O mapa criado pelo modelo apresentou comportamento similar ao *ground truth* e o *dead reckoning*, durante algumas etapas houve reconhecimento de pistas visuais mantendo

Tabela 4 – Parâmetros utilizados no modelo

Place Cell	
Neurônio	50×50
Distância entre os Neurônios	$1.3\ m$
σ_p	3.5
k_1	0.001
k_2	0.001
τ	1
η	0.9
ϕ_0	50000
ϕ_1	100000
w_{inh}	0.1
C_0	$\phi_0/Neurônio$
C_1	$\phi_1/160000$
Head Direction Cell	
Neurônio	8
σ_{hd}	20
Resolução	$45\ \circ$
Pistas visuais	
Threshold	150
Número de grupos BOW	700

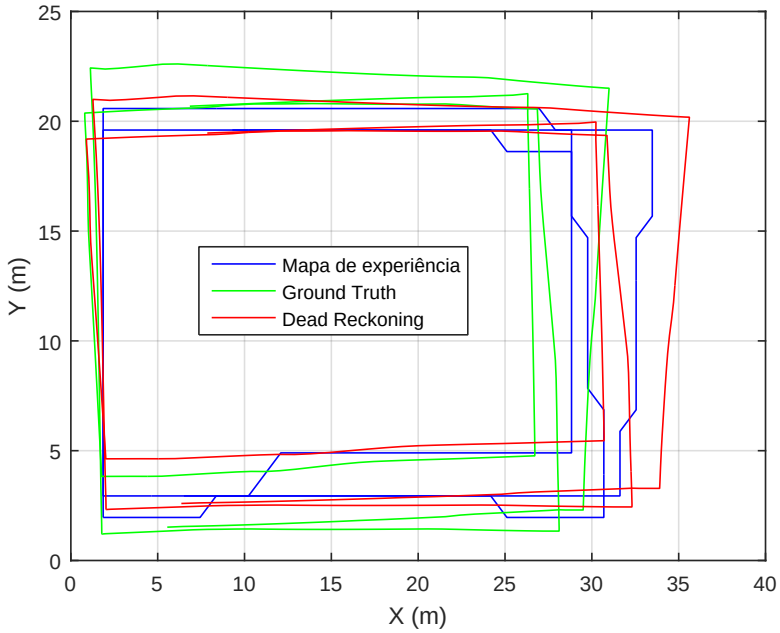


Figura 49 – Trajetória calculada durante o experimento simulado. Em verde a trajetória real do robô, em vermelho a trajetória criada através da odometria do robô, e em azul o mapa criado utilizando o modelo proposto.

a ativação dos neurônios em neurônios previamente visitados. Nas etapas que não ocorreu o reconhecimento, as ligações idiotéticas do modelo modificaram a ativação dos neurônios na direção que a correnteza estava influenciando modelo.

Na Figura 50 são apresentados os neurônios mais ativos na rede durante o experimento.

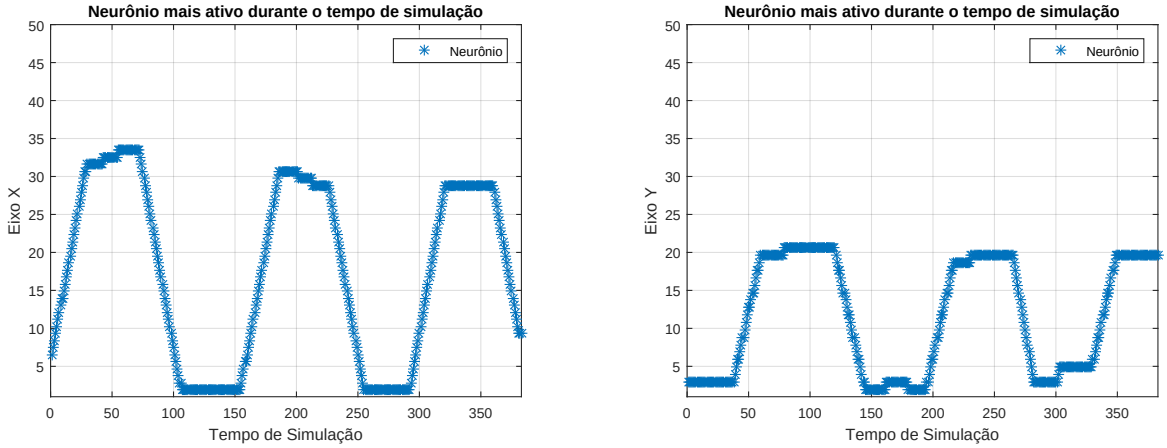


Figura 50 – Ativação dos neurônios durante o tempo de simulação.

A ativação dos neurônios apresentada na Figura 50, mostra que mesmo para distancias longas a rede neural proposta não perdeu o seu padrão de ativação. Também, devido à trajetória proposta os padrões de ativação são semelhantes. Na Figura 51 é apresentado o erro de localização do modelo e do *dead reckoning* em relação ao *ground truth*.

O erro encontrado pelo modelo foi menor quando comparado com o *dead reckoning* em quase todo o experimento, e novamente o comportamento ruidoso do modelo quando comparado ao *ground truth* e a curva do *dead reckoning* acontece devido ao *dead reckoning* ser criado através do *ground truth*.

Comparativos com a versão anterior do DolphinSLAM não foram realizados devido às características do algoritmo. Uma vez que, quando não ocorre o reconhecimento de um local previamente visitado, o mapa de experiência gerado é igual ao mapa gerado pelo *dead-reckoning*. Sendo assim, os resultados obtidos pelo modelo proposto apresentaram a modificação da ativação da rede neural e consequentemente um mapa mais realístico quando comparado ao *ground-truth* mesmo na ausência do reconhecimento das pistas visuais.

Neste capítulo foi apresentado o treinamento da rede para o reconhecimento da correnteza. Após isso foram apresentados três datasets utilizados para a validação do mesmo. O primeiro dataset foi focado somente no movimento do robô através da cor-

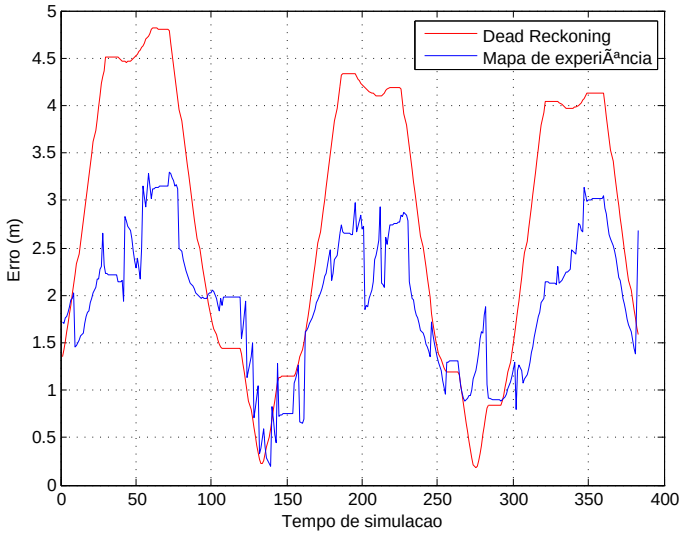


Figura 51 – Erro calculado entre o mapa de experiência e *Dead Reckoning* em relação ao *Ground Truth*.

renteza, encontrando resultados melhores que o *dead reckoning*. O segundo e terceiro apresentaram trajetórias diferentes mas utilizaram o mesmo cenário, porém diferente da primeira abordagem. Os resultados encontrados foram obtidos a partir dos mapas mais similares ao *ground truth* quando comparados aos mapas do *dead reckoning*. No próximo capítulo serão apresentadas as conclusões deste trabalho e os trabalhos futuros.

7 Considerações Finais

Neste trabalho foi apresentado um modelo bioinspirado para SLAM subaquático que busca incorporar o tratamento da correnteza marítima no modelo da rede neural. Para tal, utiliza-se a estrutura do algoritmo DolphinSLAM (SILVEIRA et al., 2015) e os modelos de localização propostos por Stringer et al. (2002b) e Stringer et al. (2002a)

Primeiramente foi apresentada a formulação do problema de SLAM e os principais métodos probabilísticos para a resolução do mesmo. Após essa contextualização, foi descrito um método bioinspirado baseado nas descobertas das *place cells*, *head direction cells* e *grid cells*.

Devido às características específicas do ambiente, foram apresentados os desafios encontrados no meio subaquático, e subsequentemente realizou-se um estudo dos principais sensores utilizados e o seu princípio de funcionamento. Os trabalhos relacionados a SLAM subaquático foram apresentados e na Tabela 1 categorizados por ano, robô, tipo de algoritmo e o tipo do cenário.

Após essa etapa, apresentou-se em detalhes os módulos do algoritmo bioinspirado DolphinSLAM e a abordagem proposta para esse trabalho. No capítulo do modelo proposto, foi realizada uma discussão relativa aos problemas ocasionados pela correnteza e posteriormente apresentou-se o modelo e os equacionamentos que o regem.

O treinamento da rede foi apresentado através dos dois diferente tipos de ligações que o modelo possui: ligações recorrentes e ligações idiotéticas. Nas ligações recorrentes cada *place cell* é definida como ativa e os pesos correspondentes são calculados e encontra-se um comportamento gaussiano simétrico. Este comportamento é responsável por manter a estabilidade da ativação em um mesmo neurônio quando não a informação das ligações idiotéticas.

As ligações idiotéticas foram treinadas através de todas as *place cells* do ambiente em conjunto com a informação da orientação do robô, orientação e velocidade da correnteza. O perfil dos pesos em uma área específica do modelo foi apresentado onde encontrou-se um comportamento assimétrico dos pesos o qual é responsável por deslocar o pacote de energia da rede conforme a correnteza.

Um comparativo entre os pesos das ligações recorrentes e das ligações idiotéticas das mesmas *place cells* foram apresentadas onde foi possível perceber as diferenças entre os mesmos. Também mostrou-se para a mesma região de neurônios o comportamento dos pesos para as diferentes orientações da correnteza.

No primeiro experimento, foi realizada uma simulação com o robô RexROV utili-

zando a correnteza com sentido leste para desloca-lo sem a influência do movimento dos *thrusters*. O modelo proposto apresentou um erro menor que o *Dead Reckoning* quando comparado ao *ground thruth*. Também foi mostrada a ativação dos neurônios durante o tempo do experimento, onde notou-se somente mudança na ativação nos neurônios no eixo X.

Para o segundo experimento definiu-se uma trajetória diferente realizada pelo Rex-ROV e um novo modelo de ambiente. O ambiente proposto assemelhasse a marina do Yacht clube Rio Grande onde possui barcos, píer, e estruturas desenvolvidas pelo ser humano afundadas no mar. O modelo proposto apresentou um comportamento mais parecido ao *ground truth*, incluindo um erro menor de localização na direção da correnteza. O gráfico da ativação dos neurônios foi apresentada, e durante o tempo demonstrou uma robustez do método durante movimento do robô.

No terceiro experimento utilizou-se o mesmo cenário proposto no segundo experimento com a diferença da trajetória realizada. A trajetória foi mais extensa possuindo pontos de reconhecimento das pistas visuais e o modelo apresentou resultados melhores que o *dead reckoning*. As ativações dos neurônios, o mapa criado e o erro foram apresentados.

O modelo de CANN $2D$ apresentou resultados satisfatórios durante os experimentos. Ambos os mapas calculados possuíram erros menores que o *Dead Reckoning*, e as ativações dos neurônios durante o tempo de execução foram coerentes.

Para trabalhos futuros espera-se avaliar o desempenho do modelo em datasets coletados em ambientes reais. Outro ponto a ser abordado é a extensão do modelo para ambientes $3D$ avaliando a correnteza e a estabilidade da rede proposta. Outras abordagens de redes neurais podem ser utilizadas para o aprendizado da correnteza, como o *Deep Learning*. Também avaliar a possibilidade de utilizar o modelo em veículos aéreos não tripulados (VANT), uma vez que neste ambiente existe a influência do vento.

Referências

ARTHUR, D.; VASSILVITSKII, S. k-means++: The advantages of careful seeding. In: SOCIETY FOR INDUSTRIAL AND APPLIED MATHEMATICS. *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. [S.l.], 2007. p. 1027–1035. Citado na página 62.

BAILEY, T.; DURRANT-WHYTE, H. Simultaneous localization and mapping (slam): Part ii. *IEEE Robotics & Automation Magazine*, v. 13, n. 3, p. 108–117, 2006. Citado 2 vezes nas páginas 29 e 31.

BALL, D. et al. Openratslam: an open source brain-based slam system. *Autonomous Robots*, Springer, v. 34, n. 3, p. 149–176, 2013. Citado na página 34.

BARKBY, S. et al. Bathymetric particle filter slam using trajectory maps. *The International Journal of Robotics Research*, SAGE Publications, p. 0278364912459666, 2012. Citado 2 vezes nas páginas 54 e 58.

BAY, H.; TUYTELAARS, T.; GOOL, L. V. Surf: Speeded up robust features. In: *Computer vision–ECCV 2006*. [S.l.]: Springer, 2006. p. 404–417. Citado na página 61.

BONIN-FONT, F. et al. Stereo slam for robust dense 3d reconstruction of underwater environments. In: IEEE. *OCEANS 2015-Genova*. [S.l.], 2015. p. 1–6. Citado 2 vezes nas páginas 55 e 58.

BOTELHO, S.; DREWS-JR, P.; LEIVAS, G. Nlmap - visual-based self localization and mapping for autonomous underwater vehicles. In: *IEEE OCEANS*. [S.l.: s.n.], 2008. p. 1–6. ISSN 0197-7385. Citado na página 53.

BRADSKI, G.; KAEHLER, A. *Learning OpenCV: Computer vision with the OpenCV library*. [S.l.]: "O'Reilly Media, Inc.", 2008. Citado na página 59.

BRASILIS, R. Exploração do pré-sal alavanca inovação tecnológica. *Revista Brasilis*, 2012. Citado na página 19.

BURGUERA, A.; BONIN-FONT, F.; OLIVER, G. Trajectory-based visual localization in underwater surveying missions. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 15, n. 1, p. 1708–1735, 2015. Citado 2 vezes nas páginas 56 e 58.

BURGUERA, A.; GONZÁLEZ, Y.; OLIVER, G. Underwater slam with robocentric trajectory using a mechanically scanned imaging sonar. In: IEEE. *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. [S.l.], 2011. p. 3577–3582. Citado 3 vezes nas páginas 21, 54 e 58.

BURGUERA, A.; GONZÁLEZ, Y.; OLIVER, G. The uspic: Performing scan matching localization using an imaging sonar. *Sensors*, Molecular Diversity Preservation International, v. 12, n. 6, p. 7855–7885, 2012. Citado na página 50.

CAIN, C.; LEONESSA, A. Fastslam using compressed occupancy grids. *Journal of Sensors*, Hindawi Publishing Corporation, v. 2016, 2016. Citado na página 32.

CENTENO, M. *Rovfurg-ii: Projeto e construção de um veículo subaquático não tripulado de baixo custo*. Dissertação (Mestrado), 2007. Citado na página 19.

CHAVES, S. M. et al. Opportunistic sampling-based active visual slam for underwater inspection. *Autonomous Robots*, Springer, v. 40, n. 7, p. 1245–1265, 2016. Citado 2 vezes nas páginas 57 e 58.

CHOI, J. et al. Ekf slam using acoustic sources for autonomous underwater vehicle equipped with two hydrophones. In: IEEE. *OCEANS 2016 MTS/IEEE Monterey*. [S.l.], 2016. p. 1–4. Citado 2 vezes nas páginas 57 e 58.

CODEVILLA, F. et al. Achieving turbidity robustness on underwater images local feature detection. *International journal of computer vision*, v. 60, n. 2, p. 91–110, 2014. Citado na página 39.

CSURKA, G. et al. Visual categorization with bags of keypoints. In: *In Workshop on Statistical Learning in Computer Vision, ECCV*. [S.l.: s.n.], 2004. p. 1–22. Citado na página 61.

CUMMINS, M.; NEWMAN, P. Fab-map: Probabilistic localization and mapping in the space of appearance. *The International Journal of Robotics Research*, v. 27, n. 6, p. 647–665, 2008. Disponível em: <http://ijr.sagepub.com/content/27/6/647.abstract>. Citado na página 59.

DISSANAYAKE, G. et al. The aiding of a low-cost strapdown inertial measurement unit using vehicle model constraints for land vehicle applications. *Robotics and Automation, IEEE Transactions on*, IEEE, v. 17, n. 5, p. 731–747, 2001. Citado na página 41.

DISSANAYAKE, M. G. et al. A solution to the simultaneous localization and map building (slam) problem. *IEEE Transactions on robotics and automation*, IEEE, v. 17, n. 3, p. 229–241, 2001. Citado na página 28.

DJURIC, P. M. et al. Particle filtering. *IEEE signal processing magazine*, IEEE, v. 20, n. 5, p. 19–38, 2003. Citado na página 30.

DREWS-JR, P.; BOTELHO, S.; GOMES, S. SLAM in underwater environment using SIFT and topologic maps. In: IEEE. *Robotic Symposium, 2008. LARS'08. IEEE Latin American*. [S.l.], 2008. p. 91–96. Citado 2 vezes nas páginas 53 e 58.

DREWS-JR, P. L. J. et al. Underwater depth estimation and image restoration based on single images. *IEEE Computer Graphics and Applications*, v. 36, n. 2, p. 24–35, Mar 2016. ISSN 0272-1716. Citado na página 39.

DREWS, P. et al. Transmission estimation in underwater single images. In: IEEE. *Computer Vision Workshops (ICCVW), 2013 IEEE International Conference on*. [S.l.], 2013. p. 825–830. Citado na página 39.

DUARTE, A. C. et al. Towards comparison of underwater slam methods: An open dataset collection. In: IEEE. *OCEANS 2016 MTS/IEEE Monterey*. [S.l.], 2016. p. 1–5. Citado na página 69.

DUNN, W. *Rotational vibration gyroscope*. Google Patents, 1995. US Patent 5,377,544. Disponível em: <https://www.google.com/patents/US5377544>. Citado na página 40.

DUNN, W.; ROOP, R. *Vibration monolithic gyroscope*. Google Patents, 1994. US Patent 5,329,815. Disponível em: <https://www.google.com/patents/US5329815>. Citado na página 40.

DURRANT-WHYTE, H.; BAILEY, T. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, IEEE, v. 13, n. 2, p. 99–110, 2006. Citado 4 vezes nas páginas 21, 24, 26 e 29.

ENGEL, P. M. *Módulo de Auto-Localizaçao para um Agente Exploratório usando Filtro de Kalman*. Dissertação (Mestrado) — UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL, 2003. Citado na página 27.

EUSTICE, R.; PIZARRO, O.; SINGH, H. Visually augmented navigation in an unstructured environment using a delayed state history. In: IEEE. *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*. [S.l.], 2004. v. 1, p. 25–32. Citado 3 vezes nas páginas 52, 53 e 58.

EUSTICE, R. M.; PIZARRO, O. M.; SINGH, H. Visually augmented navigation for autonomous underwater vehicles. *Oceanic Engineering, IEEE Journal of*, IEEE, v. 33, n. 2, p. 103–122, 2008. Citado 3 vezes nas páginas 21, 53 e 58.

FAIRFIELD, N.; KANTOR, G.; WETTERGREEN, D. Real-time slam with octree evidence grids for exploration in underwater tunnels. *Journal of Field Robotics*, Wiley Online Library, v. 24, n. 1-2, p. 03–21, 2007. Citado 2 vezes nas páginas 53 e 58.

FERREIRA, F. et al. Real-time optical slam-based mosaicking for unmanned underwater vehicles. *Intelligent Service Robotics*, Springer, v. 5, n. 1, p. 55–71, 2012. Citado 3 vezes nas páginas 28, 54 e 58.

FIRING, E. Acoustic doppler current profiling measurements and navigation. *WHP Off. Rep. WHP09-1, WOCE Rep. 68*, v. 91, 1991. Citado na página 75.

FOX, D. H. D.; BURGARD, W.; THRUN, S. A highly efficient fastslam algorithm for generating cyclic maps of large-scale environments from raw laser range measurements. In: *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*. [S.l.: s.n.], 2003. Citado na página 31.

FYHN, M. et al. Spatial representation in the entorhinal cortex. *Science*, American Association for the Advancement of Science, v. 305, n. 5688, p. 1258–1264, 2004. Citado na página 34.

GABRIEL, T. A. R. *Desenvolvimento de Sistema de Navegação de Baixo Custo de Veículo Terrestre Não Tripulado*. Dissertação (Mestrado), 2014. Citado na página 40.

HAFTING, T. et al. Microstructure of a spatial map in the entorhinal cortex. *Nature*, Nature Publishing Group, v. 436, n. 7052, p. 801–806, 2005. Citado 2 vezes nas páginas 21 e 34.

HOLLINGER, G. et al. Towards improved prediction of ocean processes using statistical machine learning. In: *Robotics: Science and Systems Workshop on Robotics for Environmental Monitoring*. [S.l.: s.n.], 2012. Citado na página 22.

HONG, S. et al. A robust loop-closure method for visual slam in unstructured seafloor environments. *Autonomous Robots*, Springer, p. 1–15, 2015. Citado 2 vezes nas páginas 56 e 58.

HONG, S.; KIM, T.; KIM, J. Underwater visual slam with loop-closure using image-to-image link recovery. In: IEEE. *OCEANS 2015-Genova*. [S.l.], 2015. p. 1–6. Citado 2 vezes nas páginas 56 e 58.

HU, M.-K. Visual pattern recognition by moment invariants. *IRE transactions on information theory*, IEEE, v. 8, n. 2, p. 179–187, 1962. Citado na página 61.

HURTÓS, N.; CUFI, X.; SALVI, J. Calibration of optical camera coupled to acoustic multibeam for underwater 3d scene reconstruction. In: IEEE. *OCEANS 2010 IEEE-Sydney*. [S.l.], 2010. p. 1–7. Citado na página 20.

KAESS, M.; RANGANATHAN, A.; DELLAERT, F. isam: Incremental smoothing and mapping. *Robotics, IEEE Transactions on*, IEEE, v. 24, n. 6, p. 1365–1378, 2008. Citado na página 55.

KALMAN, R. E. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, American Society of Mechanical Engineers, v. 82, n. 1, p. 35–45, 1960. Citado na página 28.

KIM, A.; EUSTICE, R. M. Real-time visual slam for autonomous underwater hull inspection using visual saliency. *Robotics, IEEE Transactions on*, IEEE, v. 29, n. 3, p. 719–733, 2013. Citado 2 vezes nas páginas 55 e 58.

KOHONEN, T. Self-organizing and maps. *Berlin, Heidelberg, New York: Springer*, v. 1997, p. 2001, 1995. Citado na página 36.

KOUZOUBOV, K.; AUSTIN, D. Hybrid topological/metric approach to slam. In: IEEE. *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*. [S.l.], 2004. v. 1, p. 872–877. Citado na página 28.

LEE, C. S. et al. Slam with sc-phd filters: An underwater vehicle application. *Robotics & Automation Magazine, IEEE*, IEEE, v. 21, n. 2, p. 38–45, 2014. Citado 2 vezes nas páginas 55 e 58.

LEFEVRE, H. C. *The fiber-optic gyroscope*. [S.l.]: Artech house, 2014. Citado na página 40.

LEONARD, J. L.; CARPENTER, R. N.; FEDER, H. J. S. Stochastic mapping using forward look sonar. *Robotica*, Cambridge Univ Press, v. 19, n. 05, p. 467–480, 2001. Citado 2 vezes nas páginas 51 e 58.

LIU, G. et al. Auv cruise path planning based on energy priority and current model. 2013. Citado na página 74.

LOWE, D. G. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, Springer, v. 60, n. 2, p. 91–110, 2004. Citado na página 61.

LUCAS, B. D.; KANADE, T. et al. An iterative image registration technique with an application to stereo vision. In: *IJCAI*. [S.l.: s.n.], 1981. v. 81, p. 674–679. Citado na página 52.

MACHADO, M. et al. A topological descriptor of acoustic images for navigation and mapping. In: *IEEE 12th Latin American Robotics Symposium LARS*. [S.l.: s.n.], 2015. p. 1–6. Citado na página 22.

MALLIOS, A. et al. Ekf-slam for auv navigation under probabilistic sonar scan-matching. In: IEEE. *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. [S.l.], 2010. p. 4404–4411. Citado 4 vezes nas páginas 28, 54, 55 e 58.

MALLIOS, A. et al. Scan matching slam in underwater environments. *Autonomous Robots*, Springer, v. 36, n. 3, p. 181–198, 2014. Citado 2 vezes nas páginas 55 e 58.

MANHÃES, M. M. M. et al. UUV simulator: A gazebo-based package for underwater intervention and multi-robot simulation. In: *OCEANS'16 MTS/IEEE Monterey*. [S.l.: s.n.], 2016. p. 1–8. Citado na página 83.

MASSOT-CAMPOS, M. et al. Submap bathymetric slam using structured light in underwater environments. In: IEEE. *Autonomous Underwater Vehicles (AUV), 2016 IEEE/OES*. [S.l.], 2016. p. 181–188. Citado 2 vezes nas páginas 56 e 58.

MEDAGODA, L. et al. Water column current profile aided localisation combined with view-based slam for autonomous underwater vehicle navigation. In: IEEE. *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. [S.l.], 2011. p. 3048–3055. Citado na página 75.

MILFORD, M.; WYETH, G. Mapping a suburb with a single camera using a biologically inspired SLAM system. *Robotics, IEEE Transactions on*, v. 24, n. 5, p. 1038–1053, Oct 2008. ISSN 1552-3098. Citado 6 vezes nas páginas 21, 28, 35, 36, 56 e 59.

MILFORD, M. J.; WYETH, G. F.; PRASSER, D. Ratslam: a hippocampal model for simultaneous localization and mapping. In: IEEE. *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*. [S.l.], 2004. v. 1, p. 403–408. Citado 2 vezes nas páginas 34 e 35.

MONTEMERLO, M. et al. Fastslam: A factored solution to the simultaneous localization and mapping problem. In: *Aaai/iaai*. [S.l.: s.n.], 2002. p. 593–598. Citado na página 30.

MONTEMERLO, M. et al. Fastslam: A factored solution to the simultaneous localization and mapping problem. In: *Eighteenth National Conference on Artificial Intelligence*. [S.l.: s.n.], 2002. p. 593–598. Citado na página 53.

MORENO, D.; BURGUERA, A.; OLIVER, G. Sss-slam: An object oriented matlab framework for underwater slam using side scan sonar. *Proceedings of Jornadas de Automática*, v. 2014, 2014. Citado 2 vezes nas páginas 55 e 58.

MOSER, E. I.; KROPFF, E.; MOSER, M.-B. Place cells, grid cells, and the brain's spatial representation system. *Neuroscience*, v. 31, n. 1, p. 69, 2008. Citado na página 34.

NAFOUKI, C.; CONRADT, J. Spatial navigation algorithms for autonomous robotics. Citeseer, 2015. Citado na página 32.

NEWMAN, P.; LEONARD, J. Pure range-only sub-sea slam. In: IEEE. *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*. [S.l.], 2003. v. 2, p. 1921–1926. Citado 2 vezes nas páginas 52 e 58.

NEWMAN, P. M.; LEONARD, J. J.; RIKOSKI, R. J. Towards constant-time slam on an autonomous underwater vehicle using synthetic aperture sonar. In: SPRINGER. *Robotics Research. The Eleventh International Symposium*. [S.l.], 2005. p. 409–420. Citado 2 vezes nas páginas 52 e 58.

NOAA. *How much of the ocean have we explored?* 2014. Disponível em: <http://http://oceanservice.noaa.gov/facts/exploration.html>. Citado na página 19.

O'KEEFE, J.; DOSTROVSKY, J. The hippocampus as a spatial map. preliminary evidence from unit activity in the freely-moving rat. *Brain research*, Elsevier, v. 34, n. 1, p. 171–175, 1971. Citado 2 vezes nas páginas 21 e 32.

O'KEEFE, J.; NADEL, L. *The hippocampus as a cognitive map*. [S.l.]: Oxford University Press, USA, 1978. Citado na página 32.

OLSON, E.; LEONARD, J. J.; TELLER, S. Robust range-only beacon localization. *Oceanic Engineering, IEEE Journal of*, IEEE, v. 31, n. 4, p. 949–958, 2006. Citado 2 vezes nas páginas 53 e 58.

PALOMER, A.; RIDAO, P.; RIBAS, D. Multibeam 3d underwater slam with probabilistic registration. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 16, n. 4, p. 560, 2016. Citado 2 vezes nas páginas 56 e 58.

PAULL, L. et al. AUV navigation and localization: A review. *IEEE Journal of Oceanic Engineering*, v. 39, n. 1, p. 131–149, Jan 2014. ISSN 0364-9059. Citado na página 47.

PEDLOSKY, J. *Ocean circulation theory*. [S.l.]: Springer Science & Business Media, 2013. Citado na página 74.

PETRES, C. et al. Path planning for autonomous underwater vehicles. *IEEE Transactions on Robotics*, IEEE, v. 23, n. 2, p. 331–341, 2007. Citado na página 21.

POST, E. J. Sagnac effect. *Reviews of Modern Physics*, APS, v. 39, n. 2, p. 475, 1967. Citado na página 40.

PRATS, M. et al. An open source tool for simulation and supervision of underwater intervention missions. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. [S.l.: s.n.], 2012. p. 2577–2582. Citado 2 vezes nas páginas 56 e 61.

QUIGLEY, M. et al. ROS: an open-source robot operating system. In: *IEEE ICRA - Workshop on Open Source Software*. [S.l.: s.n.], 2009. Citado 2 vezes nas páginas 56 e 59.

RIBAS, D. et al. Underwater slam in man-made structured environments. *Journal of Field Robotics*, Wiley Online Library, v. 25, n. 11-12, p. 898–921, 2008. Citado 2 vezes nas páginas 54 e 58.

RIBEIRO, F. J. L. *Sistema de Monitoramento Subaquático Para Exploração de Petróleo Usando Redes de Sensores Acústicos*. Tese (Doutorado) — Universidade Federal do Rio de Janeiro, 2012. Citado na página 19.

RIBERIO, R. de O. *Filtragem de Partículas na Estimação dos Parâmetros de Canais Rádio Móvel*. Dissertação (Mestrado) — Univesidade Estadual de Londrina, 2012. Citado na página 30.

ROMAN, C.; SINGH, H. Improved vehicle based multibeam bathymetry using sub-maps and slam. In: IEEE. *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*. [S.l.], 2005. p. 3662–3669. Citado 2 vezes nas páginas 52 e 58.

RUIZ, I. T. et al. Feature extraction and data association for auv concurrent mapping and localisation. In: IEEE. *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*. [S.l.], 2001. v. 3, p. 2785–2790. Citado 2 vezes nas páginas 51 e 58.

RUIZ, I. T. et al. Concurrent mapping and localization using sidescan sonar. *Oceanic Engineering, IEEE Journal of*, IEEE, v. 29, n. 2, p. 442–456, 2004. Citado 2 vezes nas páginas 52 e 58.

SÁEZ, J. M. et al. Underwater 3d slam through entropy minimization. In: IEEE. *Robotics and automation, 2006. ICRA 2006. Proceedings 2006 IEEE international conference on*. [S.l.], 2006. p. 3562–3567. Citado 2 vezes nas páginas 53 e 58.

SANTOS, M. M. dos. *Descrição e detecção de regiões subaquáticas parcialmente estruturadas em imagens acústicas adquiridas por um sonar de imageamento frontal*. Dissertação (Mestrado) — Universidade Federal do Rio Grande, 2016. Citado 3 vezes nas páginas 22, 60 e 61.

SANTOS, M. M. dos et al. A topological descriptor of acoustic images for navigation and mapping. In: IEEE. *2015 12th Latin American Robotics Symposium and 2015 3rd Brazilian Symposium on Robotics (LARS-SBR)*. [S.l.], 2015. p. 289–294. Citado 2 vezes nas páginas 50 e 61.

SANTOS, M. M. dos et al. A modified topological descriptor for forward looking sonar images. In: IEEE. *OCEANS 2016-Shanghai*. [S.l.], 2016. p. 1–5. Citado na página 22.

SCHÄLING, B. *The boost C++ libraries*. [S.l.]: Boris Schäling, 2011. Citado na página 59.

SELVATICI, A. H. P. *Construçao de Mapas de Objetos para Navegaçao de Robôs*. Tese (Doutorado) — Universidade de São Paulo, 2009. Citado na página 27.

SILVEIRA, L. *Sistema Bioinspirado para Mapeamento e Localização de Robôs Móveis em Ambientes Subaquáticos*. Dissertação (Mestrado) — Universidade Federal do Rio Grande, 2015. Citado 11 vezes nas páginas 21, 22, 26, 28, 47, 60, 62, 63, 64, 65 e 68.

SILVEIRA, L. et al. An open-source bio-inspired solution to underwater SLAM. In: *4th IFAC Workshop on Navigation, Guidance and Control of Underwater Vehicles (NGCUV) 2015*. [S.l.: s.n.], 2015. v. 48, n. 2, p. 212–217. Citado 6 vezes nas páginas 21, 28, 56, 58, 59 e 95.

SMITH, R. C.; CHEESEMAN, P. On the representation and estimation of spatial uncertainty. *The international journal of Robotics Research*, Sage Publications, v. 5, n. 4, p. 56–68, 1986. Citado na página 28.

SNYDER, J. Doppler velocity log (dvl) navigation for observation-class rovs. In: IEEE. *OCEANS 2010*. [S.l.], 2010. p. 1–9. Citado na página 42.

STANWAY, M. J. Water profile navigation with an acoustic doppler current profiler. In: IEEE. *OCEANS 2010 IEEE-Sydney*. [S.l.], 2010. p. 1–5. Citado na página 75.

STRINGER, S. et al. Self-organizing continuous attractor networks and path integration: two-dimensional models of place cells. *Network: Computation in Neural Systems*, Taylor & Francis, v. 13, n. 4, p. 429–446, 2002. Citado 5 vezes nas páginas 59, 64, 69, 70 e 95.

STRINGER, S. et al. Self-organizing continuous attractor networks and path integration: one-dimensional models of head direction cells. *Network: Computation in Neural Systems*, Taylor & Francis, v. 13, n. 2, p. 217–242, 2002. Citado 4 vezes nas páginas 69, 70, 72 e 95.

TAUBE, J. S.; MULLER, R. U.; RANCK, J. B. Head-direction cells recorded from the postsubiculum in freely moving rats. i. description and quantitative analysis. *The Journal of neuroscience*, Soc Neuroscience, v. 10, n. 2, p. 420–435, 1990. Citado 2 vezes nas páginas 21 e 33.

THRUN, S. A probabilistic on-line mapping algorithm for teams of mobile robots. *The International Journal of Robotics Research*, SAGE Publications, v. 20, n. 5, p. 335–363, 2001. Citado na página 28.

THRUN, S.; LEONARD, J. J. Simultaneous localization and mapping. In: _____. *Springer Handbook of Robotics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p. 871–889. ISBN 978-3-540-30301-5. Disponível em: <http://dx.doi.org/10.1007/978-3-540-30301-5_38>. Citado 5 vezes nas páginas 26, 27, 28, 30 e 31.

THRUN, S. et al. FastSLAM: an efficient solution to the simultaneous localization and mapping problem with unknown data association. *Journal of Machine Learning Research*, v. 4, n. 3, p. 380–407, 2004. Citado na página 21.

WALTER, M.; HOVER, F.; LEONARD, J. Slam for ship hull inspection using exactly sparse extended information filters. In: IEEE. *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*. [S.l.], 2008. p. 1463–1470. Citado 2 vezes nas páginas 54 e 58.

WATANABE, Y.; OCHI, H. et al. A study of inverse ssbl acoustic positioning with data transmission for multiple auv navigation. In: IEEE. *2012 Oceans-Yeosu*. [S.l.], 2012. p. 1–6. Citado na página 45.

WILLIAMS, S.; MAHON, I. Simultaneous localisation and mapping on the great barrier reef. In: IEEE. *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*. [S.l.], 2004. v. 2, p. 1771–1776. Citado 2 vezes nas páginas 52 e 58.

WILLIAMS, S. B. et al. Autonomous underwater simultaneous localisation and map building. In: IEEE. *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*. [S.l.], 2000. v. 2, p. 1793–1798. Citado 2 vezes nas páginas 51 e 58.

WYNN, R. B. et al. Autonomous underwater vehicles (auvs): Their past, present and future contributions to the advancement of marine geoscience. *Marine Geology*, Elsevier, v. 352, p. 451–468, 2014. Citado na página 20.

YARTSEV, M. M.; ULANOVSKY, N. Representation of three-dimensional space in the hippocampus of flying bats. *Science*, American Association for the Advancement of Science, v. 340, n. 6130, p. 367–372, 2013. Citado 2 vezes nas páginas 59 e 64.

YOKOZUKA, M.; MATSUMOTO, O. Sub-map dividing and realignment fastslam by blocking gibbs mcem for large-scale 3-d grid mapping. *Advanced Robotics*, Taylor & Francis, v. 26, n. 14, p. 1649–1675, 2012. Citado na página 32.

YUH, J. Design and control of autonomous underwater robots: A survey. *Autonomous Robots*, v. 8, n. 1, p. 7–24, 2000. ISSN 1573-7527. Disponível em: <http://dx.doi.org/10.1023/A:1008984701078>. Citado na página 20.

ZAFFARI, G. B. et al. Exploring the dolphinslam's parameters. In: IEEE. *OCEANS 2016-Shanghai*. [S.l.], 2016. p. 1–5. Citado 3 vezes nas páginas 61, 62 e 69.